This Research was Sponsored by

THE NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D. C.

Research Grant NASA No. NsG 36-60

A PROGRAMMABLE DIGITAL COMPENSATOR FOR SINGLE LOOP HIGH PERFORMANCE DIGITAL SERVOMECHANISMS

Report No. EDC 7-65-30

by

Donald J. Gawlowicz

1965

Harry W. Mergler
Professor of Engineering
Co-investigator

Digital
Systems
Engineering

ABSTRACT

16417

The purpose of this research was to investigate the feasibility of designing a programmable digital compensator of sufficiently low cost to be employed in a single high performance control loop with virtually any analog dynamics in the frequency range of interest. A compensator with a cost comparable to that of components presently used in high performance servomechanisms was designed and built. The machine was wired to solve linear, constant coefficient, difference equations. The output of the machine is therefore a weighted sum of past inputs and outputs of the machine. The inputs and outputs are stored in a magnetostrictive delay line which is clocked at one megacycle. The desired program is selected by inserting the appropriate weighting constants into a programmable diode matrix. Up to fifteen past inputs and outputs may be used in the program. The time between input samples may also be programmed within reasonable limits.

The compensator was tested with three different forms of analog dynamics within the two to twenty cycle per second recommended frequency range of the compensator. Significant improvements in performance were obtained in all cases. These test results, the complete design details of the compensator, and the sampled-data synthesis procedures that were employed are presented.

Author

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to his advisor, Professor H. W. Mergler, of Case Institute's Engineering Design Center, for his support and encouragement. The help of my wife in the final preparation of this document is gratefully acknowledged. Thanks are also due the National Aeronautical and Space Administration for their sponsorship of this research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

<u>Real and Complex Symbols</u>

a_i	Weighting constant for e_{n-i}
A	A constant proportional to the analog gain, K_A
b_i	Weighting constant for y_{n-i}
c(t)	Output of the closed loop at time t
c_n	Output of the closed loop at time nT
c_{SMAX}	Maximum closed loop output with a unit step input
D(z)	Compensator's z- transfer function
e_{n-i}	Error at time (n-i)T
E(z)	z- transform of the sampled error sequence
F(z)	z- transfer function corresponding to P(s)/s
$F_\Delta(z)$	Advanced z- transfer function corresponding to P(s)/s
AG(z)	z- transfer function of the entire forward control loop excluding the compensator
J	Upper summation limit in the closed loop z- transfer function
j	Lowest power of z^{-1} in the numerator of G(z)
K	Gain setting
K_A	Analog gain
K_D	Digital gain
K_i	Constants programmed in the diode matrix
K(z)	Closed loop z- transfer function

M	Number of past outputs, y_{n-i}, in the compensator's linear program
N	Number of error terms, less one, in the compensator's linear program
P(s)	Laplace transform of the analog plant
p	Number of iterations of the basic computation cycle which have been programmed, p = 1, 2, . . . , 8
q	Order of the highest order input polynomial the closed loop is designed to follow
r(t)	Input to the closed loop at time t
r_n	Input to the closed loop at time nT
s	Complex variable in the Laplace transform
T	Sample time
T_R	Time required to read one word out of memory
u_i	Constant in a factor of the numerator of G(z)
X_{n-i}	Represents either an input or an output of the compensator at time (n-i)T
y_{n-i}	Output of the compensator at time (n-i)T
Y(z)	z- transform of the compensator's output sequence
z	Complex variable in the z- transform
α	Equals β + j
α_i	Product of a_i and A
β	The fraction of a sample time by which the closed loop output leads the closed loop input
γ_i	Weighting constant for r_{n-j-i} (coefficient in the closed loop z- transfer function)
Δ	Parameter in the advanced z- transform
ζ	Damping ratio of a second order linear system
ω_n	Natural frequency of a second order linear system

Registers

rA	BDC Accumulator in arithmetic logic
rA_R	Consists of the cells of rA which define $\|y_n\|$ and all of the cells of rA to the right of these cells
rB	Shift register operated as a ring counter in the control logic
rE	Output register of the analog to digital converter
rH	Input register of the zero-order hold
rM	Binary counter in the control logic; also serves as the scalor for the operational multiplier
rM_R	Consists of the seven cells of rM which lie furthest to the right
rW	Binary counter in control logic
rY	Consists of the eight cells of rA which define the compensator output, y_n

Boolean Variables

A_e	True when rA_R equals zero
A_i	Output of $C_i(A)$ taken from a small isolating resistor; $i=0, 1, \ldots, 11$
B	True when rE is to be counted backward
B_A	True when rA is to be counted backward
$C_i(X)$	i^1 th cell of rX, where rX is any register
CLEAR	Switches to "zero" when rH is to be cleared
C_L	True when rA is to be cleared
C_M	Master clock (1 mc)
D_i	Outputs of sample time programming switch; $i=0, 1, 2$
E_e	True when rE equals zero
F_i	Inputs to logic which generates A_e; $i=1, \ldots, 5$
F	True when rE is to be counted forward
F_A	True when rA is to be counted backward
f	Four megacycle clock (derived from crystal oscillator)
G	True when rB is to revert to state $S_o(B)$
I_o	Input to cell $C_0(B)$
I_1	Input to cell $C_1(B)$
I_{SP}	Input to flip-flop S_p
P_{03}	Three microsecond pulse which switches to "one" when rM overflows
P_{04}	Four microsecond pulse which switches to "one" when rM overflows
P_x	Eighty nanosecond pulse which switches to "one" when x switches to "one"; x is any Boolean variable

R	Memory output flip-flop
R_E	True while rE is being read
R_H	Executes the transfer $+127 \rightarrow rH$
S_A	Sign bit of rA; true when sign is negative
S_E	Sign bit of rE; true when sign is negative
S_H	Executes the transfer $-127 \rightarrow rH$
$S_i(X)$	State i of rX, where rX is any register
S_K	Sign bit of weighting constant
S_P	Flip-flop which holds the sign of the product of a weighting constant and input or output of the compensator; true when sign is negative
T_{Ai}	Trigger inputs to cells $C_i(A)$; $i=0, 1, \ldots, 5$
T_L	True when analog to digital converter is to track the analog input
TRANSFER	Switches to "zero" when the transfer $rY \rightarrow rH$ is to be executed
W	Input to the delay line

ABBREVIATIONS

A/D	Analog to Digital
BDC	Bidirectional Counter
D/A	Digital to Analog
DDA	Digital Differential Analyzer
LSB	Least Significant Bit

LOGICAL CONNECTIVES

U	Logical "OR"
$\oplus$	Exclusive "OR"
$\cdot$	Logical "AND" (also without the dot)

CHAPTER ONE
INTRODUCTION TO DIGITAL COMPENSATION

1.1 Introduction

In the design of large digital control systems, two basic approaches are currently in use. One approach is to use one large centralized digital machine to perform all of the required computations. The second approach is to use many smaller, special purpose machines to perform the required functions.

The first approach is highly attractive since it allows the greatest time sharing of equipment and should therefore result in the simplest overall system. The problems involved in realizing the full potential of this approach, however, are enormous because of the wide range of computational tasks that are usually present in a given system. An inertial guidance system, for example, must indicate the position and velocity of a vehicle (navigate), and control these quantities in a closed loop fashion so that the body follows a prescribed trajectory. The navigation problem requires high accuracy and low speed while the control problem demands high speed with less accuracy. As a result, a digital computer designed to handle both problems is very likely to have an accuracy-bandwidth product far exceeding that required for either problem.[15]

A serious fault of the centralized approach is that a failure in the centralized computer can result in the failure of a major part of the entire system. In many fields efforts are therefore being made to develop the full potential of the decentralized approach. For example, avionics systems have been designed with centralized computers, but heavy emphasis is now being placed upon the development of decentralized systems so that an equipment failure is unlikely to result in the failure of an entire mission.

Although the centralized approach will probably yield the simplest, most reliable digital systems, many authorities believe that in the meantime, excellent progress can be made by following the decentralized approach.[17] It is with this philosophy in mind that this research was performed.

1.2 Digital Control

In recent years a large amount of engineering effort has been devoted to the design and development of digital automatic control systems. This activity has resulted largely because of two important advantages of the digital control system:

1. Compatibility with the digital computer, and
2. An increase in accuracy over analog techniques.

The first attempts at employing the large data handling and storage capabilities of the digital computer in the field of automatic control often involved the placement of digital to analog and analog to digital converters between the computer and an analog control loop. In addition to the disadvantage of having additional hardware, these converters usually resulted in a loss of accuracy. With a digital control loop, direct communication with the digital computer is possible since the command and feedback data are numerically represented in the loop.

The accuracy of an analog control system is determined by the precision with which the amplitude of electronic signals can be represented and detected. Therefore, the accuracy is ultimately limited by the stability and noise characteristics of the electronic components that are employed. In the digital system, where the command and feedback data are represented and compared numerically, an increase in accuracy is obtained by the addition of logical elements. Since these elements operate at signal levels far in excess of the noise and drift levels of the electronic components that are used, the accuracy of a digital control system is not limited by the accuracy of the electronic components.

A digital control system may be classified as being either incremental or absolute. In an incremental system, a given change in the output variable is specified by an electronic pulse. In an absolute system, the value of the output variable is represented by a coded number.

An incremental control loop is shown in Figure 1.1. The incremental encoder, or quantizer, generates a pulse each time a given increment of change in the controlled variable occurs.

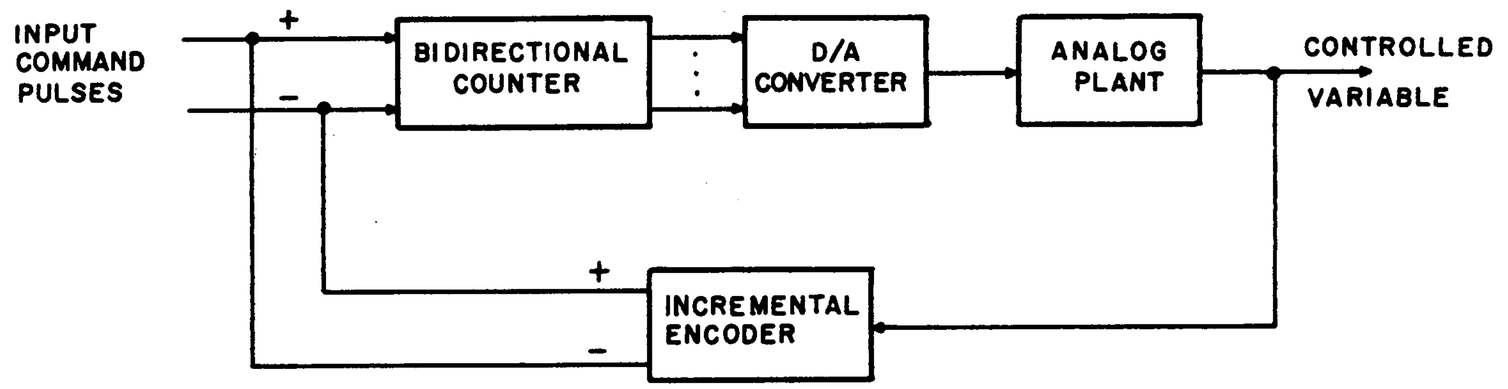


FIGURE 1.1— INCREMENTAL DIGITAL CONTROL LOOP

The pulse occurs on one of two output lines, depending upon the direction in which the controlled variable has moved. These feed-pulses are subtracted from the input command pulses so that the number stored in the counter represents the error between the desired and the actual value of the controlled variable. This error is then converted to an analog signal to control the analog plant. The error is continually updated as feedback pulses occur so that the error is not sampled, even though it is quantized.

An absolute digital control loop is shown in Figure 1. 2. The output of the absolute encoder is a coded number equal to the value of the controlled variable. This number is subtracted from the input command, which is also a coded number, to generate the error. This error is then used to drive the analog plant as before. Although this system could also be instrumented to update the error each time a feedback change occurred, the error is usually computed at a fixed frequency in order to simplify the electronics. Therefore, the error in an absolute digital loop is usually both sampled and quantized.

The incremental system is inferior to the absolute digital control system because it is possible to lose the point of reference for the controlled variable. If a malfunction results in an incorrect error signal in an incremental system, this error will remain in the system forever. However, even with this serious disadvantage, incremental systems are far more widely

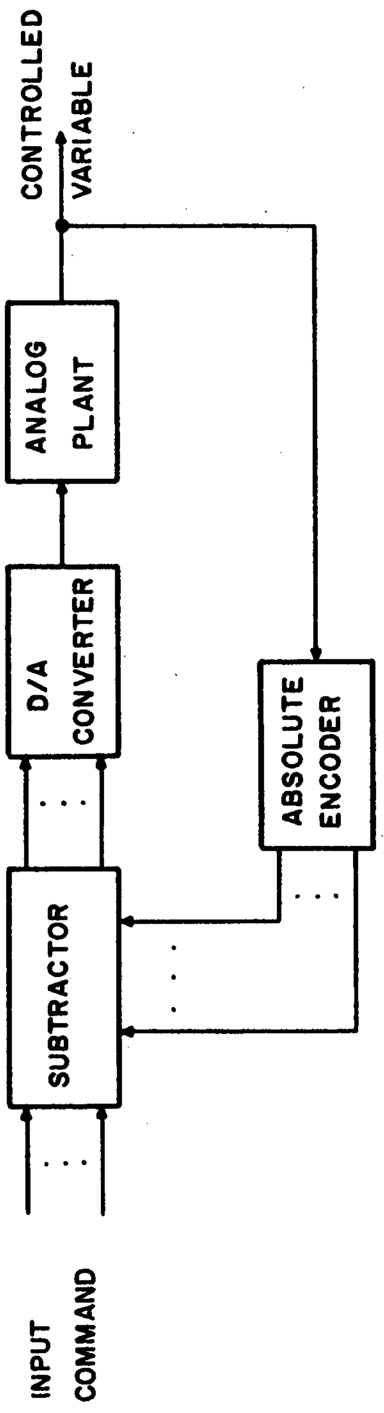


FIGURE I.2— ABSOLUTE DIGITAL CONTROL LOOP

used than absolute systems because they result in the simplest hardware.

The accuracy of an unloaded digital control system, whether it is incremental or absolute, is limited by two factors:

1. The resolution of the feedback encoder, and
2. The null stability of the digital to analog converter in the forward path of the loop.

Progress in both of these areas has resulted in digital control loops which exceed the accuracy of the best possible analog systems by more than an order of magnitude.

1.3 Digital Control Applications

The use of digital control systems is limited to applications where the increased cost and complexity are offset by the advantages described in Section 1.2. For this reason these control systems have been used mainly in high performance positioning servomechanisms. In most of these applications, the definition of high performance includes high static accuracy (within four quantum), wide bandwidth (ten cycles per second), and compatibility with the digital computer.

In numerically controlled machine tools, for example, the need for high accuracy is obvious, and a wide bandwidth is always desired in order to reduce machining time. Also, it is usually necessary to provide some combination of off-line and real time digital computational capability in order to automate the burden-task of transforming engineering drawings into actual command information for the servos.[12, 22] It is therefore highly efficient, if not necessary, to use servos which can communicate directly with this equipment.

Digital control loops are also now widely used for the gyro loops in inertial guidance platforms. Again, the need for high accuracy and the desirability of having direct communication links with the digital guidance computer, has dictated the use of digital control systems.

These two examples of digital control applications have been included in order to clearly define the class of control systems which are under consideration in this thesis. The control problem usually involves three or fewer control loops which are part of a much larger overall system.

1.4 Stability of Digital Control Systems

We have discussed in Section 1.2 the main reasons why digital control systems are extensively employed. However, the introduction of digital equipment into the control loop has increased the problem of obtaining a satisfactory response. The controlled variable tends to exhibit a ripple in the steady state, and, more important, stability in the large and a good transient response are difficult to obtain.

In an incremental system, the steady state ripple occurs because the analog dynamics is driven by a quantized signal.

Since the error signal required to balance a given output load will, in practice, never exactly equal an integral number of quanta, the error must oscillate with an amplitude of at least one quantum so that the average error equals the signal required to support the load. If this oscillation is not observed in practice, it is because friction or some other nonlinearity in the null supplies an additional force to balance the load. In an absolute digital system, ripple is introduced by the sampling process as well as by the quantization.[21, 24]

The large scale stability and transient response are also adversely affected by the sampling and quantizing processes because the loop is closed only at the instants in time at which the error is updated.

1.5 Compensation of Single Loop Digital Servomechanisms

As with any servomechanism, a digital control loop may be compensated by changing the analog dynamics. However, it is usually impractical to alter the analog plant, and a compensation network cannot be located in the feedback loop because of the presence of the encoder. The compensator must therefore be located between the digital to analog converter and the analog plant. This results in some serious problems:

1. High power linear amplification will usually be required to drive the analog plant. This is especially objectionable with a DC system since it may be

impossible to maintain the null stability that can be obtained by driving the analog plant directly with the digital to analog converter.

2. With an AC system, there is always the problem of designing a suitable AC compensator.
3. If, as is usually the case, the compensator must supply some derivative control, any noise spikes present in the digital to analog converter output are accentuated.
4. If a digital stepping motor is used, it is impossible to place a compensator after the digital to analog converter since a digital to analog converter does not exist as a distinct element in the loop; the conversion is actually performed by the motor.

We are therefore led to consider the possibility of compensating the system with information that is still in digital form.

Besides the disadvantages involved with employing analog compensation, we can list several important advantages that can be obtained by employing digital control:

1. The digital computer program can be easily synthesized to obtain closed loop control systems to meet a wide variety of performance criteria, which include integration, prediction, noise reduction, and no error response to polynomial inputs after a finite settling time.
2. The above responses can be obtained regardless of the analog dynamics in the loop and without any realizability constraints placed upon the compensator program other than assuring that inputs precede outputs.
3. The system can be made adaptive, since the computational algorithm of the compensator, or the constants within a given algorithm, can be conveniently changed as operating conditions change.
4. Any degree of computational accuracy is possible.

These advantages result from the extreme flexibility of a digital machine in handling complex computations. This flexibility is obtained, of course, at the expense of additional hardware.

Digital machines which have been previously used to compensate digital servomechanisms fall into one of the following categories:

1. The compensator is of a special type to provide a particular mode of control with a given type of system.[2, 9, 19, 23, 30, 31]
2. The compensation is performed by a computer which is also assigned to perform numerous other tasks.[13]
3. The compensation is performed by a computer which is capable of compensating a large number of control loops.[14]

Special purpose compensators will always find widespread application because of their simplicity and because they eliminate the problems associated with analog compensation. They lack the flexibility, however, to realize the many advantages that can be obtained by employing digital techniques. While machines in the second category have this flexibility, there are many small scale applications where they cannot be economically employed. They also lie outside the design philosophy we have elected to follow.

Machines in the third category also have the required flexibility but are too large and expensive to be used for the very common problem of compensating three or fewer loops.

The purpose of this thesis was to attempt to design a digital machine which would fill the gap between the highly specialized type of digital compensators and the large scale digital machines which have previously been used.

1.6 The Single Loop Digital Compensator

The decision was made to concentrate on the compensation of a single control loop, for if an efficient solution to this problem could be obtained, then one or more loops could be compensated by using individual compensators for each loop. Also, it was felt that the experience gained in the solution of this problem would be valuable in obtaining a more efficient solution to the more common two or three loop problem.

It was decided that the single loop digital compensator should have:

1. the ability to be programmed to compensate a wide variety of analog dynamics with different performance criteria,
2. sufficient bandwidth to compensate a digital control loop capable of following a large signal ten cycle per second input, and,
3. a cost comparable to the high quality servo components that are used in digital control systems.

In order to satisfy the third restriction, it was necessary to wire a given computational algorithm into the compensator. Programming is accomplished by changing the constants in this algorithm. In the prototype machine these constants are stored in a read only memory whose contents can be conveniently altered manually but which cannot be changed by electrical noise. If ordinary fixed constant compensation is required in a specific application, this memory could be replaced by a permanent read only memory. Alternatively, the compensation could be adaptively changed upon command by another digital computer provided this memory was replaced by a read-write memory.

The function of the compensator is to operate upon the sampled error signal and generate the number sequence which will cause the analog plant to react in a prescribed manner. The location of the compensator is shown in Figure 1. 3. The time, T, between successive readings of the error is constant and is called the sample time. The compensator reads the error, e_n, at time nT and emits an output, y_n, at time nT, where n is an integer. If the analog dynamics is linear, then the variety of closed loop responses mentioned in the previous section can be obtained if y_n is defined by the following linear, constant coefficient difference equation:[21]

$$y_n = \sum_{i=1}^{M} b_i \; y_{n-i} + \sum_{i=0}^{N} a_i \; e_{n-i} \; . \qquad (1.1)$$

The compensator was therefore wired to solve this equation. M is an integer larger than zero and N is an integer larger than or equal to zero. The present output of the compensator is therefore equal to a weighted sum of M previous outputs of the

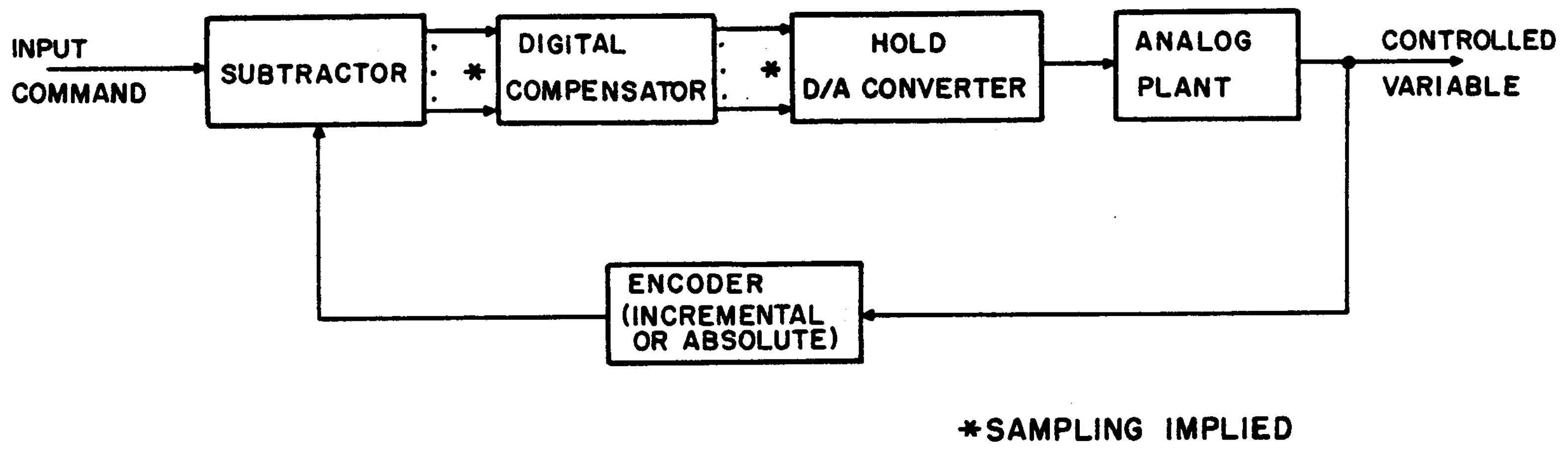


FIGURE I.3— ERROR COMPENSATED DIGITAL CONTROL LOOP

compensator, plus a weighted sum of its present input and N past inputs.

We can begin to appreciate the full power of this computational algorithm by relating equation (1.1) to some of the familiar control algorithms that are employed in analog control loops. For example, suppose we wish the output of the compensator to be equal to a weighted sum of the error and the derivative of the error. If the derivative is approximated by $(e_n - e_{n-1})/T$, we obtain:

$$y_n = K_p\ e_n + K_d \left[\frac{e_n - e_{n-1}}{T}\right] \qquad (1.2)$$

$$= (K_p + K_d/T)\ e_n - (K_d/T)\ e_{n-1}\ ,$$

where K_p and K_d are gain constants. This equation is a digital equivalent of the familiar proportional plus derivative form of control commonly found in analog servomechanisms. Comparing with equation (1.1), we see that $a_o = K_p + K_d/T$ and $a_1 = - K_d/T$. All the other constants are zero.

If we also wish the output of the compensator to be partially composed of the integral of the error, we can approximate the integral by $y_{n-1} + T\ e_n$ to obtain:

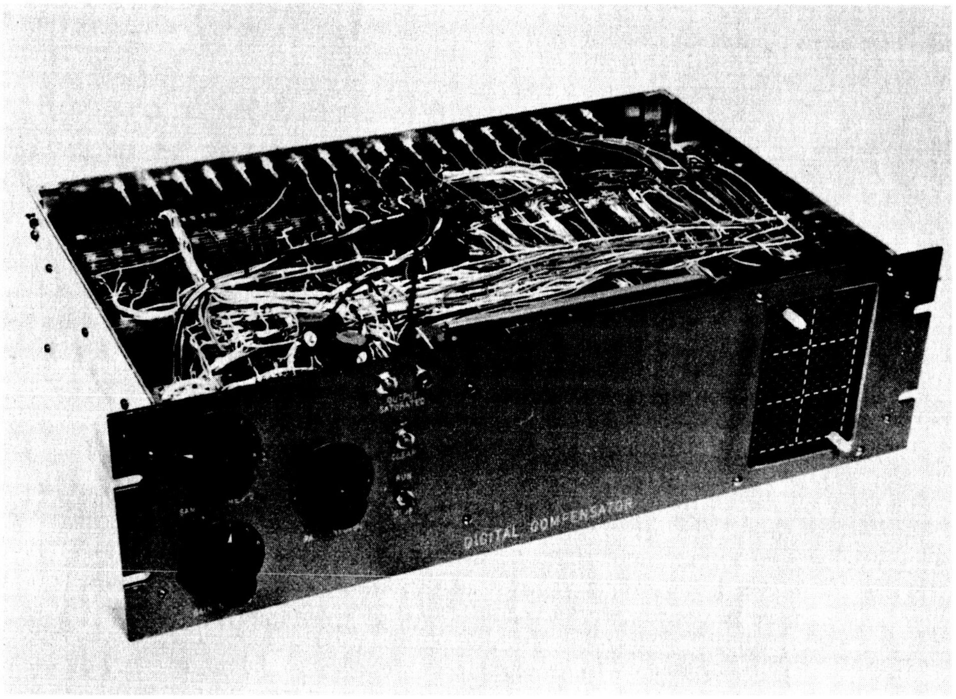
$$y_n = K_I \; (y_{n-1} + T\, e_n) + K_p\, e_n + K_d \frac{e_n - e_{n-1}}{T} \qquad (1.3)$$

$$= K_I \; y_{n-1} + (K_I\, T + K_p + K_d/T)\, e_n - (K_d/T) e_{n-1} \;,$$

where K_I is the integral gain constant. We now have $b_1 = K_I$, $a_o = K_I T + K_p + K_d/T$, and $a_1 = - K_d/T$. This equation represents a digital equivalent of the proportional plus integral plus derivative control algorithm often found in analog control loops.

In general, the constants a_i; $i = 0, 1, \ldots, N$, and b_i; $i = 1, 2, \ldots, M$, can be selected so that equation (1.1) will perform the functions of integration, differentiation, smoothing, prediction, amplification, and any combination thereof. However, we shall not revert to these basic operations in order to synthesize the constants $\{a_i\}$ and $\{b_j\}$. The manner in which these constants are selected, and the wide variety of closed loop responses which can be obtained, are described in Section 2.1 and Chapters Four and Five. If one wishes to obtain a stronger appreciation of the capabilities of the algorithm defined by equation (1.1), it may be desirable to read these sections at this time. In passing, we might mention that this algorithm is often referred to as a linear computer program, and a digital machine which solves equation (1.1) is commonly called a linear discrete filter.

The next chapter describes some of the major design decisions that were made in arriving at an efficient design for the compensator. Photographs of the compensator are shown in Figure 1.4.



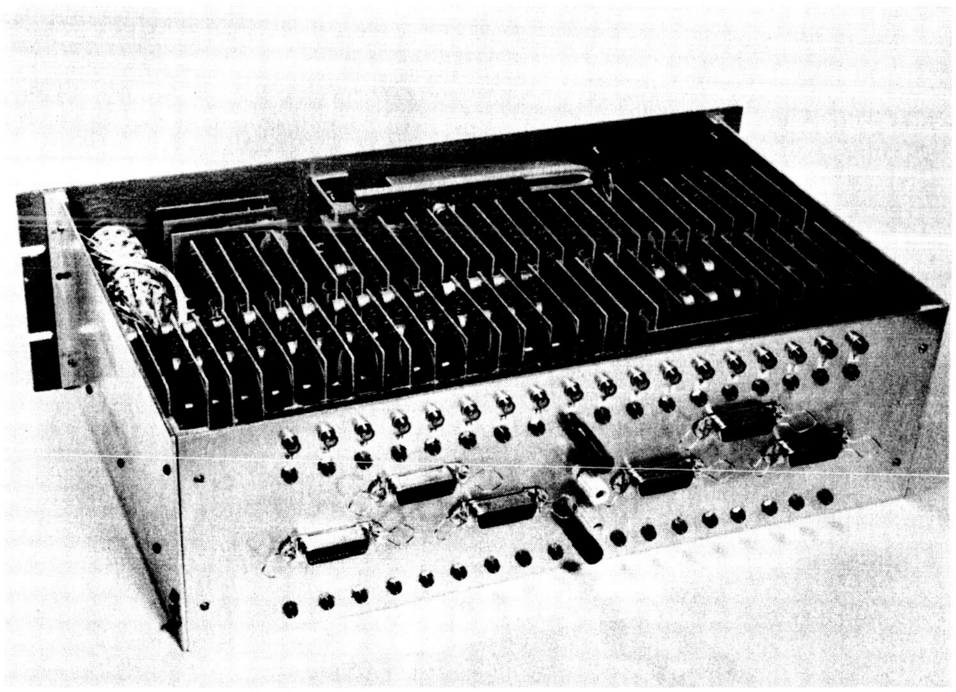


FIGURE 1.4– PHOTOGRAPHS OF THE DIGITAL COMPENSATOR

CHAPTER TWO
MAJOR DESIGN CONSIDERATIONS

2.1 The Scaling Problem

The first step in the design process was to determine the range of the constants $\{a_i\}$ and $\{b_j\}$ in equation (1.1). The constants $\{a_i\}$ are strongly dependent upon the analog gain in the system, as is easily shown.

If quantization is ignored, the control loop can be modeled by the sampled-data system shown in Figure 2.1. All samplers, both real and fictitious are synchronous and are assumed to close simultaneously at a fixed frequency of period T. The fictitious samplers, which are shown by dotted lines, merely serve to emphasize that the indicated z- transforms relate the number sequences that would be obtained by sampling the corresponding analog variables in synchronism with the actual sampling.

As explained in Section 3.2, the digital compensator can be modeled by a machine which computes equation (1.1) and a computation delay which must be lumped with the analog dynamics. Any time associated with the sampling process may be included in the computation delay. We can therefore assume that the samplers remain closed for zero time, as must be the case if the z-transform approach is to yield exact results.

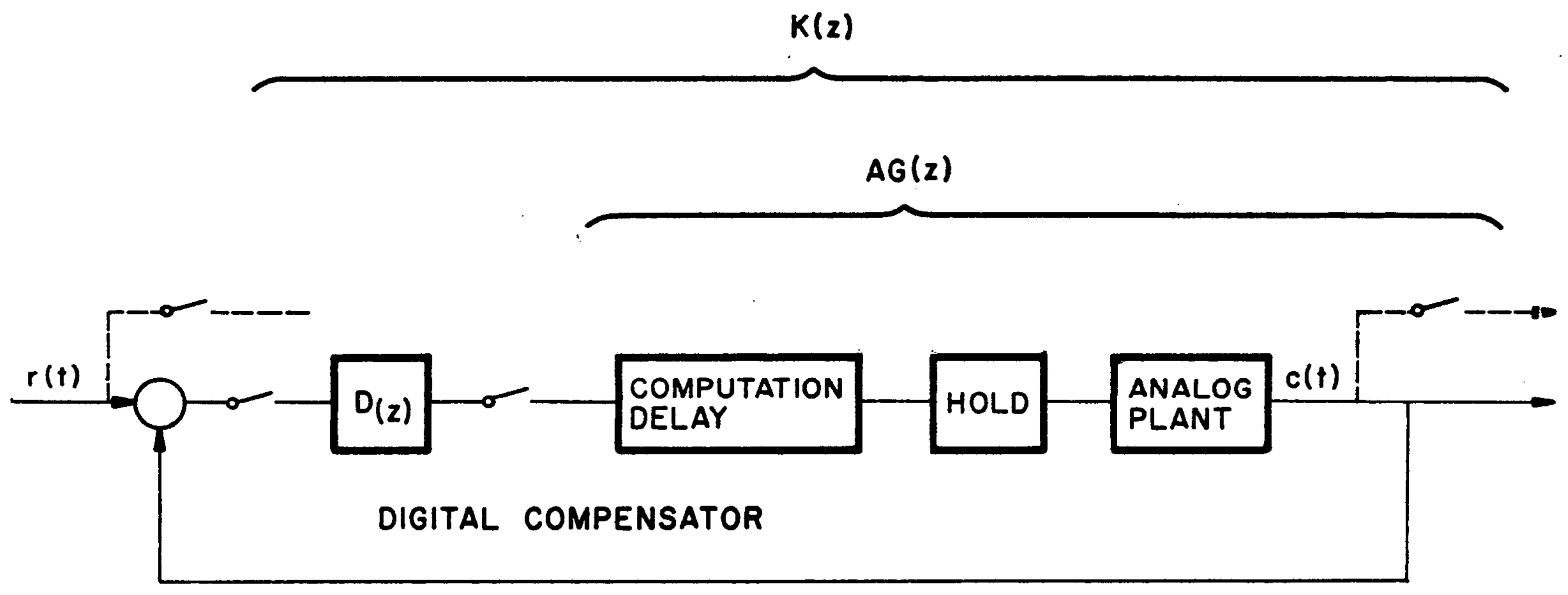


FIGURE 2.1- SAMPLED-DATA MODEL OF AN ERROR COMPENSATED DIGITAL CONTROL LOOP

Since the ideal (zero computation time) compensator solves equation (2. 1), it can be characterized by a pulse transfer function, D(z), which is obtained by z- transforming equation (1. 1):[21, 24]

$$\frac{Y(z)}{E(z)} = D(z) = \frac{\sum_{i=0}^{N} a_i z^{-i}}{1 - \sum_{i=1}^{M} b_i z^{-i}} \quad . \tag{2. 1}$$

Y(z) and E(z) are the z- transforms of the number sequences $\{y_n\}$ and $\{e_n\}$ respectively. If the analog plant is linear, the remaining dynamics in the forward path of the loop can also be defined by a pulse transfer function which consists of a ratio of polynomials in z^{-1}:[21, 24]

$$\frac{C(z)}{Y(z)} = AG(z) = A \frac{z^{-j}\left[1 + \sum_{i=1}^{Q} A_i z^{-i}\right]}{1 + \sum_{i=1}^{L} B_i z^{-i}} \quad . \tag{2. 2}$$

Q and L are integers larger than zero and j is an integer larger than or equal to zero. C(z) is the z- transform of the sequence $\{C_n\}$. If we select the closed loop transfer function, K(z), to yield a desired closed loop response, we can compute the compensator transfer function which will yield this response from the relation between K(z) and AG(z):[21, 24]

$$D(z) = \frac{K(z)}{AG(z)\left[1 - K(z)\right]} \quad , \tag{2. 3}$$

K(z) is always taken to be of the form:

$$K(z) = z^{-j} \left[\frac{\sum_{i=0}^{J} \gamma_i z^{-i}}{1 + \sum_{i=1}^{P} d_i z^{-i}} \right] \tag{2.4}$$

since this yields the computer program which utilizes the most recent error sample and hence minimizes the time delay in the loop. Again, J is an integer larger than or equal to zero and P is an integer larger than zero. The compensator transfer function is therefore given by:

$$D(z) = \frac{\sum_{i=0}^{N} \frac{\alpha_i}{A} z^{-i}}{1 - \sum_{i=1}^{M} b_i z^{-i}} . \tag{2.5}$$

The α_i's are found by multiplying the numerator of $z^j K(z)$ by the denominator of G(z), and the b_i's are obtained by multiplying the numerator of $z^j G(z)$ by the numerator of 1 - K(z). Comparison of equations (2.1) and (2.5) shows that:

$$a_i = \frac{\alpha_i}{A} . \tag{2.6}$$

A is proportional to the analog gain in the loop and the constants $\{\alpha_i\}$ are independent of the analog gain. We can therefore restrict the magnitude of the a_i's if we are free to select the analog gain. The compensator was therefore designed with the assumption that the analog gain could be selected to be any real positive number in order to avoid the necessity of using floating point notation to represent the a_i's. This does not really represent a compromise since we are merely assuming that we are capable of assigning the required analog weight to the least significant bit of the compensator output. The only function the exponent in a floating point notation would have is to define which scale of a rather involved digital to analog converter should be used, i. e., it would define the proper analog gain. One cannot reduce the analog gain requirements by providing a large digital gain. A digital gain merely means that the least significant bit at the compensator's output must be assigned a larger analog weight than the least significant bit at the input.

The approximate range of the constants $\{b_i\}$ was obtained by numerically calculating these constants for various types of analog systems in various frequency ranges with the aid of a digital computer. It was found that the magnitude of these constants was almost always smaller than four, so that these constants could also be represented with fixed point notation.

2.2 Sample Time Considerations

A sample frequency of four to eight times the closed loop bandwidth is often quoted in the literature as adequate for computer control applications.[5,28] This corresponds to a sample time of 12.5 to 25 milliseconds for a ten cycle per second servo.

This, however, is a limiting figure that is made without mention of the ripple or the error that can be expected when the control loop is operated at the system bandwidth. It was the author's opinion that a ten cycle per second input could not possibly be followed with the high precision and low ripple that is required in many applications if the input was sampled only four to eight times during one cycle. It was felt that up to 25 samples during one cycle was more realistic. This set a speed requirement for the compensator of one complete computation in four milliseconds. The compensator was actually designed with a cycle time of two milliseconds, since this was more convenient for the memory which was used. The compensator may therefore be used in closed loops with up to 120 cycles per second bandwidth in some applications, although a 20 cycle per second bandwidth is probably more reasonable in precision applications.

Sampled-data theory places no restriction upon the selection of the sample time. Computer programs can be synthesized, in theory, to provide a finite settling time equal to a fixed number of sample times regardless of the length of the sample time. This would imply that one could obtain as fast a response as one desired, regardless of the analog dynamics, by merely reducing the sample time. However, as the sample time is reduced, the loop gain must be increased; the analog plant must be driven harder in order to respond in the shorter time. A practical limit is reached when the analog plant begins to saturate. One will usually find that a settling time much smaller than the dominant time constant of the analog plant will be difficult or impossible to obtain.

Another restriction is placed upon the sample time by stability and ripple considerations. As the sample time becomes large, the sampled-data system tends to go unstable and exhibit a large ripple. It would therefore appear that one should select the sample time as small as possible in order to control the ripple, and then allow a large number of sample times for the system to settle in order to avoid overdriving the plant. However, as the settling time to sample time ratio is increased, sampled-data theory demands that the memory capacity of the compensator be increased. This is illustrated by equation (2. 3). If K(z) is selected to be a polynomial in z^{-1} with a finite number of terms, as is usually done in order to obtain a finite settling time with polynomial inputs, the number of terms in K(z) increases as the number of samples required for the system to settle is increased. The numerator and denominator polynomials of D(z) therefore become longer, which means that more past inputs and outputs of the compensator must be remembered. Although one could always abandon the sampled-data approach and approximate the compensator by a continuous element for synthesis purposes, the ability to synthesize programs capable of providing the flexibility described in Section 1. 5 would then be lost.

A sample time of 1/4 to 1/8 of the dominant time constant of the analog plant was considered sufficiently small to contain the ripple within reasonable bounds. With a sample time in this range, a settling time approximately equal to the dominant time constant of the analog plant can usually be obtained without fear of saturating the plant provided the compensator has a memory of approximately fifteen words.

A sample time of four milliseconds for a ten cycle per second bandwidth was established independent of the open loop dynamics. If this sample time is to equal 1/4 to 1/8 of the dominant time constant of the analog plant, it follows that the first break frequency of the plant must lie between five and ten cycles per second. It should not be earth-shattering news to learn that a ten cycle per second servo should be used in a ten cycle per second loop! This does not mean that slower analog systems cannot be used. It merely means that practical difficulties may be encountered in instrumenting the required driving capability.

In order to compensate lower frequency loops, it is necessary to use a lower sampling frequency in order to maintain reasonable loop gains. Provisions for increasing the sample time was therefore incorporated into the compensator. The sample time may be increased from two to approximately sixteen milliseconds in seven equal increments. The compensator is therefore capable of compensating high performance servomechanisms with

2 to 20 cycle per second bandwidths. Although lower frequency loops could always be compensated by further reducing the sample time, or by employing a continuous approximation, the inefficiency of unnecessarily using high speed computation makes this application unattractive.

2.3 Coding

In order to obtain the required logical efficiency, natural binary coding was used throughout the machine with one exception as explained in Section 3.5.

2.4 Word Length

The word length used to represent the input error does not depend upon any accuracy requirements. A large input word length in a properly compensated control loop does not enhance accuracy since the error is always close to zero when the loop is operated within its bandwidth. If the error never exceeds eight quanta, the error is only encoded with an accuracy of one part in eight regardless of the length of the input word! Indeed, this presents a very perplexing problem. The null accuracy of a digital compensator cannot be increased by an increase in word length because the added significant bits will not be used by the compensator.

The input word length is determined solely by the number of quanta error one wishes to allow before saturating the compensator. The saturation point, in turn, is determined by the desired response when the closed loop is subjected to large inputs which contain frequency components exceeding the loop bandwidth; a compensator which can accept a 128 quanta error will yield a superior response to a 128 quanta step input than a compensator which can only accept a 64 quanta error.

In ordinary error-driven digital loops, six to ten bits are typically used to encode the error. This range was used as a guide in selecting the word length for the compensator, with the final decision based upon the degree of simplification that could be attained by reducing the word length. A word length of eight bits for the input was selected as an efficient compromise. For convenience, the output word length was also taken to be eight bits.

The constants in equation (1. 1) need not be represented with high accuracy since the analog dynamics are usually not known with more than five percent accuracy, and the error in the null (where accuracy is most important) is only represented by a few significant bits. Six bits, including sign should therefore be adequate for representing the constants. However, with the instrumentation that was employed, the extra hardware required for an eight bit representation was insignificant. Eight bits were therefore used as insurance. The smaller constants could then be represented with greater accuracy and the larger constants could be represented with accuracy approaching 1 part in 128.

2.5 Incremental vs. Whole Word Computation

Numerous papers have been written concerning the relative merits of incremental and whole word computers in real-time control applications.[5, 7, 8, 26, 27, 28] Nevertheless, we must still review this issue in the light of our design problem.

If we restrict M + N in equation (1.1) to a reasonable number, let us say fourteen, then the compensator must perform fifteen multiplications and fourteen additions during each computation cycle. It was decided in Section 2.2 that a computation cycle must be completed in four milliseconds. Word size was set at eight bits including sign in Section 2.4.

A serial incremental machine with circuits in the 100 KC range would be capable of meeting the above requirements. With a whole word machine one would either have to go to higher speed circuitry or parallel operation. The incremental machine is capable of the higher iteration rate because it only computes the change that occurs in a given variable from one iteration to the next, while the whole word machine must compute a completely new value. Assume, for example, that we wish to compute the product of a constant, c, and a variable, E. The simplest type of incremental machine, the DDA (Digital Differential Analyzer),[10] is designed on the principle that the value of the variable during the present iteration, E_n, does not differ from the value of the variable during the previous iteration, E_{n-1}, by more than one quantum. Therefore, in order to form the product cE_n, the DDA need only add (or subtract) the constant c to (or from) cE_{n-1}, if the variable E has changed since the previous iteration. If E hasn't changed, then the DDA need do nothing. The DDA can therefore perform the multiplication in one add time. The simplicity and speed of the incremental machine are therefore obtained by severely limiting the amount that the input and computed variables can change during one cycle. This is a severe disadvantage in high speed real-time control applications. The output of the compensator must change far more than one quantum in four milliseconds if we wish to follow a ten cycle per second input of any reasonable amplitude. Also, if any reasonable response is expected with inputs which exceed the ten cycle per second range, such as with step inputs, both the input and output of the compensator must be allowed to change by large amounts from one computation cycle to the next. We can therefore say that for real time control applications, a statement of the computational frequency of a digital machine is completely meaningless unless accompanied by a statement defining the amount the input and output variables can change during one computation cycle.

The time required by an incremental machine to move from whatever state it is in to a new state demanded by a sudden change in input variables is called the slewing time of the machine.[5,28] This term is usually not used in conjunction with whole word machines because the slewing time of every whole word machine is one computation period.

Since the slewing time of a DDA is far too large for most real-time applications, many variations of the DDA have been designed and built in order to reduce the slewing time. Two methods have been used:

1. The increment size of a variable is increased as the change in the variable during one cycle becomes large.[5, 27, 28]
2. The machine is designed so that changes in the variables during one cycle of more than one increment are processed.[14]

In the first method accuracy is sacrificed when variable changes are large, and the computation frequency is reduced since old and new values of the variables must be compared. The slewing time of the whole word machine may be approached, but the cost of a whole word machine is reached long before the slewing time of the machine is approached.[28] The second technique, when carried to a limit, evolves into a whole word machine.[14]

Because compensator performance suitable for all ten cycle per second applications was desired, the slewing capability of a whole word machine was considered essential. Since previous efforts had shown that this objective cannot be economically attained with incremental machines, the obvious course of action was to attempt to design a whole word machine with a sufficiently high iteration rate.

It was felt that a high speed serial whole word machine could provide an economical solution for two reasons. First, the logical economy of a DDA is effected mainly by the arrangement of data in a serial memory (magnetic drum) so that each item in storage is accessible at the time it is required by the computation. This same economy can be realized with a whole word machine when the computational algorithm is restricted to equation (1. 1).[6] Second, high speed serial machines are now both practical and economical. At the present time, the cost of highly reliable discrete semiconductors suitable for use in ten megacycle circuits cost only twice as much as the 200 KC variety, and recent advances in integrated circuit technology indicate that, within a few years, ten megacycle integrated logic modules will be available for little more than contemporary transistors.[11] Any reasons for restricting designs to low frequency circuits are therefore rapidly disappearing.

The slewing time of the DDA could also be reduced to acceptable limits with high speed circuitry by merely increasing the iteration rate. This could also lead to a highly efficient design for high speed control applications. All of the logical simplicity of present DDA's would remain intact; one would merely have to trade the magnetic drum for a delay line and use high speed circuitry. If this approach was followed, however, the compensator would appear as a continuous element in the control loop, because in order to program equation (1.1) on a DDA, the DDA must generate an output after each iteration.[21] The serial, whole word design was pursued because of a reluctance to leave the sampled-data approach.

CHAPTER THREE
SIMPLIFIED TIMING AND ORGANIZATION OF THE DIGITAL COMPENSATOR

3.1 Notation and Definitions

The operation of any sequential machine may be described by a sequence of register transfers where a register is defined as an ordered collection of Boolean variables. However, this inclusive a definition is not required here. In this paper a register is defined as an ordered collection of two or more binary cells or binary memory elements. When not explicitly stated otherwise, the memory elements of a given register will be numbered from left to right, starting with zero. The name of each register will have the prefix r to distinguish registers from single Boolean variables.

In the course of describing any sequential machine, one will usually encounter Boolean variables which are functions of the state of some register, rX. One such function which is very common in the control logic of a sequential machine is defined as follows: $S_i(X) = 1$ if and only if rX is in state i. In order to simplify notation, we shall also refer to state i of rX as $S_i(X)$. The precise meaning should be clear from the context of the statements.

Another function worthy of special definition is defined as follows: $C_i(X) = 1$ if and only if the ith memory element of rX is in state 1. Also, in order to simplify notation, $C_i(X)$ will often be used as the name of the ith cell of rX.

When we are dealing with a single memory element, we shall also only use one symbol for the name of the element and for the function which assumes values equal to the state of the memory element.

3.2 Sequencing the Variables (Delay Line Timing)

The most significant logical simplification that is obtained by restricting the computational algorithm of the digital compensator to a linear program is that a sequential access memory may be used without sacrificing computational speed. The sequential nature of the computation allows one to write information into memory in such a manner that it appears at the read head exactly at the time it is needed. The obvious selection for a serial memory with the required capacity was a delay line.

The location of past inputs and outputs in the delay line, the manner in which new information is added to the memory, and the manner in which time indexing is accomplished is shown in Figure 3.1. Information is shown shifting from left to right. For clarity, the special case of a linear program containing seven input samples is shown (N=6) and five output samples (M=5) is shown.

															Control State	Mult. by
y_{n-2}	y_{n-3}	y_{n-4}	y_{n-M}	y_{n-M-1}	e_{n-1}	e_{n-2}	e_{n-3}	e_{n-4}	e_{n-5}	e_{n-6}	e_{n-7}	e_{n-8}	e_{n-9}	e_{n-10}	S_0 (B)	0
y_{n-1}	y_{n-2}												e_{n-8}	e_{n-9}	S_1 (B)	0
e_{n-9}	y_{n-1}												e_{n-7}	e_{n-8}	S_2 (B)	0
e_{n-8}	e_{n-9}												e_{n-N}	e_{n-7}	S_3 (B)	0
e_{n-7}	e_{n-8}												e_{n-5}	e_{n-N}	$S_{15-(M+N)}$ (B)	a_N
e_{n-N}	e_{n-7}												e_{n-4}	e_{n-5}	S_5 (B)	a_5
e_{n-5}	e_{n-N}												e_{n-3}	e_{n-4}	S_6 (B)	a_4
e_{n-4}	e_{n-5}												e_{n-2}	e_{n-3}	S_7 (B)	a_3
e_{n-3}	e_{n-4}												e_{n-1}	e_{n-2}	S_8 (B)	a_2
e_{n-2}	e_{n-3}												y_{n-M-1}	e_{n-1}	S_9 (B)	a_1
e_{n-1}	e_{n-2}												y_{n-M}	y_{n-M-1}	S_{15-M} (B)	a_0
e_n	e_{n-1}												y_{n-4}	y_{n-M}	S_{16-M} (B)	b_M
y_{n-M}	e_n												y_{n-3}	y_{n-4}	S_{12} (B)	b_4
y_{n-4}	y_{n-M}												y_{n-2}	y_{n-3}	S_{13} (B)	b_3
y_{n-3}	y_{n-4}												y_{n-1}	y_{n-2}	S_{14} (B)	b_2
y_{n-2}	y_{n-3}												e_{n-9}	y_{n-1}	S_{15} (B)	b_1

$N = 6, \quad M = 5$

FIGURE 3.1—DELAY LINE TIMING CHART

The delay line holds fifteen words; each word consists of one sample of either the input (e_{n-i}) or the output (y_{n-j}) of the compensator. Note that the constants associated with the computer program are not stored in the serial memory as is usually done with magnetic drum real-time computers.[13, 14] This was done for two reasons:

1. The constants must be stored permanently. If they were stored in the delay line, they would be susceptible to alteration by noise. One short transient could then permanently change the linear program.
2. Control logic for loading constants into the delay line is eliminated.

Each line in Figure 3. 1 illustrates the state of the delay line at a particular instant of time. Each line is labeled with a state of control register, rB. This register switches to state $S_i(B)$ at the instant that the delay line reaches the state illustrated by the line labeled $S_i(B)$ and remains in state $S_i(B)$ until the next illustrated state of the memory is reached. For example, the memory reaches the state shown in the first line just as the control register switches to state $S_o(B)$. rB remains in this state until the second line is reached, at which time it changes to state $S_1(B)$.

Just before the control register switches to state $S_o(B)$, a computation has been completed and the output, y_n, has been transferred from the accumulator (rA) to the hold register. The word which emerges from the delay line when $S_o(B) = 1$ is an old error term which no longer is used in the computation. This word is discarded as the compensator output is inserted into memory. This output is now called y_{n-1} since it will be used to compute the new output, y_n. rA is then cleared to zero.

In general, old error terms which are not used in the computation will emerge from memory during the next few control states. These words are multiplied by zero and recirculated. (This processing yields the simplest logic.) The computation does not actually start until state $S_{15-(M+N)}(B)$ is reached. While control is in this state the oldest error term to be used, e_{n-N}, emerges from memory and is multiplied by a_N. The product is generated in an incremental manner. As each bit emerges from memory, the increment it contributes to the product is computed and added to rA. This enables each bit to be immediately recirculated and eliminates the need for a buffer register between memory and the arithmetic logic.

The more recent error terms are then processed in the same manner (oldest terms first) until state $S_{15-M}(B)$ is reached. At this time rA has accumulated

$$\sum_{i=1}^{N} a_i \, e_{n-i} \ .$$

During state $S_{15-M}(B)$, an old output term which is no longer used in the computation leaves the delay line and is discarded. The new error term, e_n, is written into memory as the product $a_o\, e_n$ is generated. Therefore, when the control register switches to state $S_{16-M}(B)$, rA has accumulated

$$\sum_{i=0}^{N} a_i\, e_{n-i} \quad ,$$

the part of the output contributed by the error terms.

During state $S_{16-M}(B)$, the oldest output value to be used in the computation emerges from the delay line, followed by the more recent outputs. The products $b_j\, y_{n-j}$ are formed as above while the outputs, y_{n-j}, are recirculated. When state $S_{15}(B)$ is left, control returns to state $S_o(B)$ with

$$\sum_{i=0}^{N} a_i\, e_{n-i} + \sum_{i=1}^{M} b_i\, y_{n-i} = y_n \qquad (1.1)$$

in rA. y_n is transferred to the hold register and a new computation cycle begins. Upon return to state $S_o(B)$, all terms are indexed back in time since a new computation is starting.

It should be clear that N and M must satisfy the relation

$$15 - (M + N) \geq 1$$

or

$$M + N \leq 14. \qquad (3.1)$$

In other words, no more than fifteen terms may be used in the linear program.

Note also that there is a significant computation delay between the time that the input is sampled and the output is generated. In the preceding discussion we have labeled the variables as though the output is generated immediately after the input is received. Equation (1. 1) therefore describes the operation of an ideal machine which has no computation delay. The operation of the compensator, however, is identical to the operation of a machine which computes the output in zero time, and then waits for $(1 - \Delta)T$ seconds before emitting the output, where $(1 - \Delta)T$ is the computation delay $(0 \leq \Delta < 1)$. The compensator may therefore be modeled as a machine which realizes equation (1. 1), followed by a delay of $(1 - \Delta)T$ seconds. The linear program may therefore be synthesized by assuming that there is no computation delay associated with the compensator, provided a time delay of $(1 - \Delta)T$ seconds is added to the analog dynamics already in the control loop.

3. 3 Varying the Sample Time

The sample time is varied by following the control sequence $S_1(B)$ through $S_{15}(B)$ for some fixed number of iterations before returning control to state $S_o(B)$. The number of iterations is programmed with a switch as described in Appendix B.

If the number of iterations of the basic computation cycle described in Section 3. 2 equals p, then the variables must be multiplied by the constants $\{a_i/p\}$ and $\{b_i/p\}$ instead of $\{a_i\}$ and $\{b_i\}$ during each iteration.

The first time that state $S_{15-M}(B)$ is encountered after leaving state $S_o(B)$, the input to the compensator is multiplied by a_o/p as before. For the remaining iterations, however, a_o/p must be multiplied by the delay line output, not the compensator's input, while the delay line output is recirculated.

Examination of Figure 3.1 will show that the sample time, T, is given by:

$$T = T_R (1 + 15p) \tag{3.2}$$

where T_R is the time required to read one word and p is the number of iterations that are programmed. For convenience, the sample time is tabulated in Table 3.1 for the number of iterations which may be programmed (T_R = 0.128 milliseconds.)

3.4 Computation Delay

We are now in a position to write an explicit equation for the computation delay. Examination of Figure 3.1 will show that

$$\text{Computation delay} = (1 - \Delta)T = T - (15\text{-}M)T_R \, . \tag{3.3}$$

$$\text{Since} \quad T = T_R (1 + 15p), \tag{3.4}$$

$$\text{we obtain} \quad (1 - \Delta)T = T \left(\frac{15p - 14 + M}{1 + 15p}\right). \tag{3.5}$$

The delay is written in terms of a parameter, Δ, where $0 \leq \Delta \leq 1$, since this is the manner in which the dead time is usually introduced when applying the advanced z- transform.

SAMPLE TIME (MILLISECONDS)	NUMBER OF ITERATIONS
2.048	1
3.968	2
5.888	3
7.808	4
9.728	5
11.648	6
13.568	7
15.488	8

TABLE 3.1– POSSIBLE SAMPLE TIMES

The overall pulse transfer function consisting of the analog plant dynamics, the zero-order hold, and the computation delay may then be obtained from tables of advanced z- transforms. The parameter, Δ, which appears in these tables is obtained from equation (3.5):

$$\Delta = \frac{15 - M}{1 + 15p} \qquad (3.6)$$

It will be recalled that M is the number of past outputs which are used in the linear program, and p is the number of iterations of the basic computation cycle which are programmed.

3.5 Arithmetic Logic

As has already been explained, division is not required of the arithmetic unit. Multiplication and addition is performed with a binary operational multiplier and bidirectional counter (BDC) which is called rA. The arrangement is shown in Figure 3.2.

The operation of the operational multiplier is described in Appendix B and elsewhere.[20] The binary multiplier accepts an input frequency, f; a number K_i, coded in natural binary; and emits a frequency, $K_i f$. If this frequency is counted over a fixed time base, a coded number proportional to K_i is obtained. If the time base is now made proportional to another number, X_{n-i}, a number proportional to $K_i X_{n-i}$ is obtained.

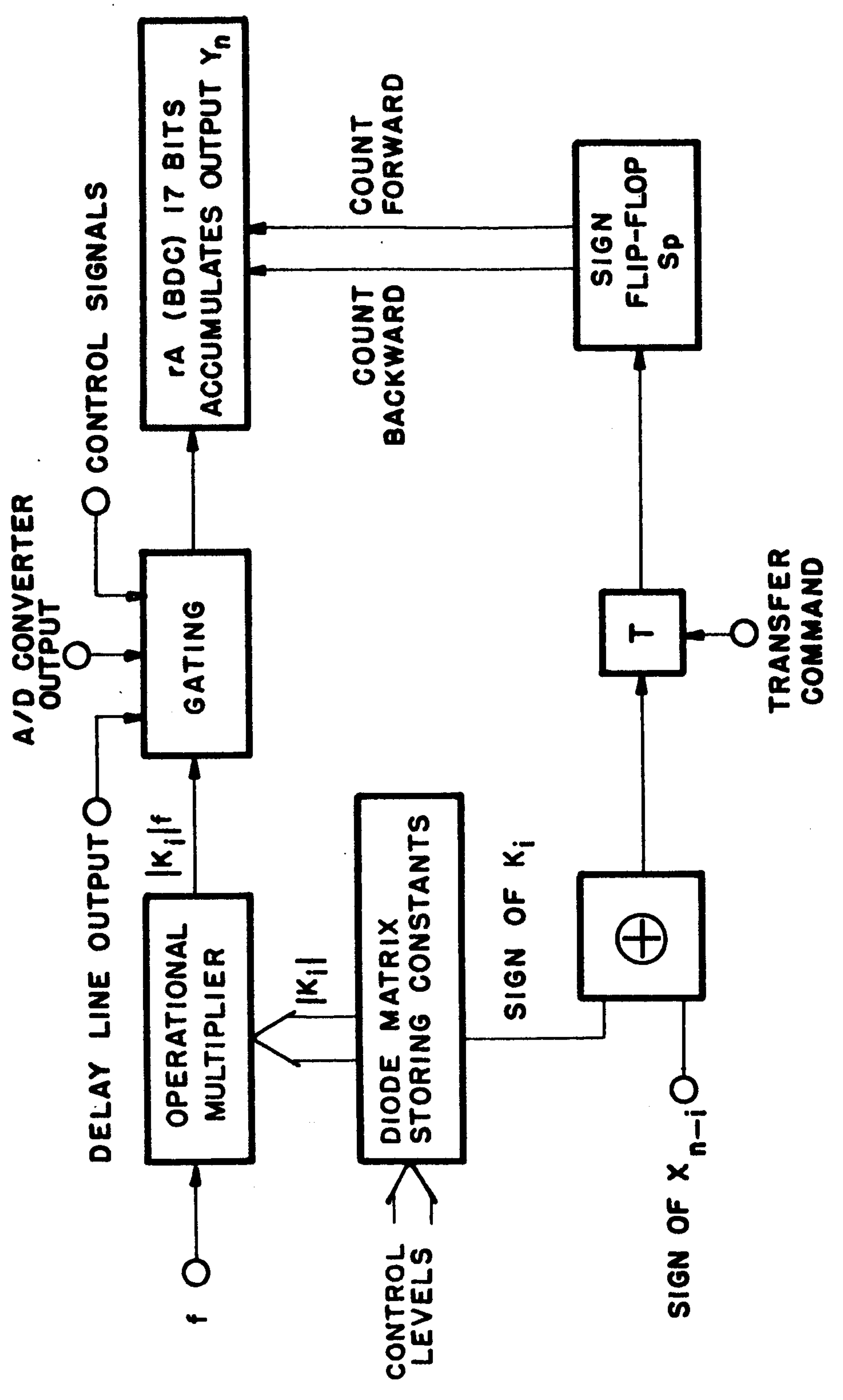


FIGURE 3.2- ARITHMETIC LOGIC

Obviously K_i is related to a weighting constant and X_{n-i} is either a past input or output of the compensator in the specific application at hand. The constants are stored in a programmable diode matrix (see Appendix B). The constants are sequentially gated into the operational multiplier so that K_i controls the output frequency of the multiplier at the time X_{n-i} is emerging from the delay line.

The time during which the output frequency is counted is controlled with very little logic because of the manner in which X_{n-i} is coded. The first bit to emerge from the delay line is the sign bit. This bit is exclusive OR'ed with the sign bit of K_i to define whether the accumulator (rA) is to count forward or backward. The remaining bits define the magnitude of X_{n-i}. A weighted code is not used. The magnitude of X_{n-i} is equal to the number of 1's in the delay line immediately following the sign bit. For example, + 60 would be stored as a "zero" followed by sixty "ones". Therefore, the required product is formed by merely counting the output of the operational multiplier during the time that "ones" are emerging from the delay line (excluding, of course, the sign bit).

One simplification which results from this instrumentation lies in the fact that the output, y_n, appears in a BDC. To insert this number into the delay line, it is only necessary to count the BDC to zero with the delay line clock as "ones" are written into the delay line. For clarity, the logic required for this operation is not illustrated in Figure 3.2.

The manner in which an input sample, e_n, must enter the machine is now clear; the error must appear in a counter which can be counted to zero as "ones" are written into memory. This counter is not shown as part of the compensator for two reasons:

1. It will usually be possible to efficiently encorporate this counter into the circuitry which forms the error in an absolute digital servo.
2. This counter does not appear in the test loop that was instrumented. The A/D converter that was used was of a type which generates the digital output in a bidirectional counter.

This method of computation offers one more simplification which is not immediately obvious. A binary counter, required in the control logic to clock the delay line, can also be used for the scalor in the binary operational multiplier. The entire arithmetic section, therefore, consists of only the accumulator, rA, and some combinational circuitry. Virtually every other multiplication scheme, whether serial or parallel, would require at least one more register in addition to an accumulator and combinational gating.

The operational multiplier can only generate a frequency which is less than the input frequency. In other words, $|K_i|$ must be less than one. If f is set equal to the delay line clock frequency, C_M, a little reflection will show that multiplication of

X_{n-i} by a constant larger than one is not possible when the smallest sample time is programmed. It was shown in Section 2.1 that the constants which weight the inputs to the compensator can always be made less than one by using a sufficiently large analog gain in the control loop. The constants which weight past outputs of the machine, however, may be larger than one. This difficulty was overcome by letting $f = 4\ C_M$. Multiplication by constants as large as four is then possible when the smallest sample time is used; larger constants are possible with larger sample times.

The frequency $4\ C_M$ (equal to four megacycles) was selected as a compromise between maximum machine capability and minimum hardware. A higher frequency would require more and faster circuits. The numerical range of the constants which would have to be programmed in order to realize the transfer functions required to compensate various types of analog systems in various frequency ranges were synthesized with the help of a digital computer; it was found that multiplication by constants larger than four would not be necessary for the vast majority of cases.

Increasing the input frequency helps to alleviate the one important disadvantage of this computational method. The output of the operational multiplier does not consist of evenly spaced pulses; it is only the average frequency over one cycle of the binary scalor within the multiplier which equals $K_i f$. The instantaneous frequency is only an approximation to $K_i f$. Since the output frequency is not accumulated for an integral number of cycles of the scalor (except for some special values of X_{n-i}) the multiplication is not exact.

It has been shown[18] that this error can be reduced if a desired output frequency, $K_i f$, is generated by first forming $2^n K_i f$ with an operational multiplier which has an input frequency of $2^n f$ and then dividing this frequency by 2^n with a mod 2^n counter. With an input frequency of 4 C_M, it is possible with most programs to generate frequencies 2^n times higher than necessary and use the first stages of rA as a mod 2^n counter to divide the frequency. The output, y_n, is then taken from the higher order stages. More will be said about this later.

3.6 Control logic

We are now in a position to understand most of the requirements of the control logic. The logic must incorporate the following circuits:

1. A counter to keep track of each bit within a word in memory.
2. A register (rB) with sixteen distinct states, $S_o(B)$, . . . , $S_{15}(B)$, to define each word.
3. A counter to control when rB is to revert back to state $S_o(B)$.
4. A precision oscillator to clock the delay line and the control circuits.

The control logic is shown in Figure 3.3. A frequency, four times larger than the delay line clock (C_M) frequency,

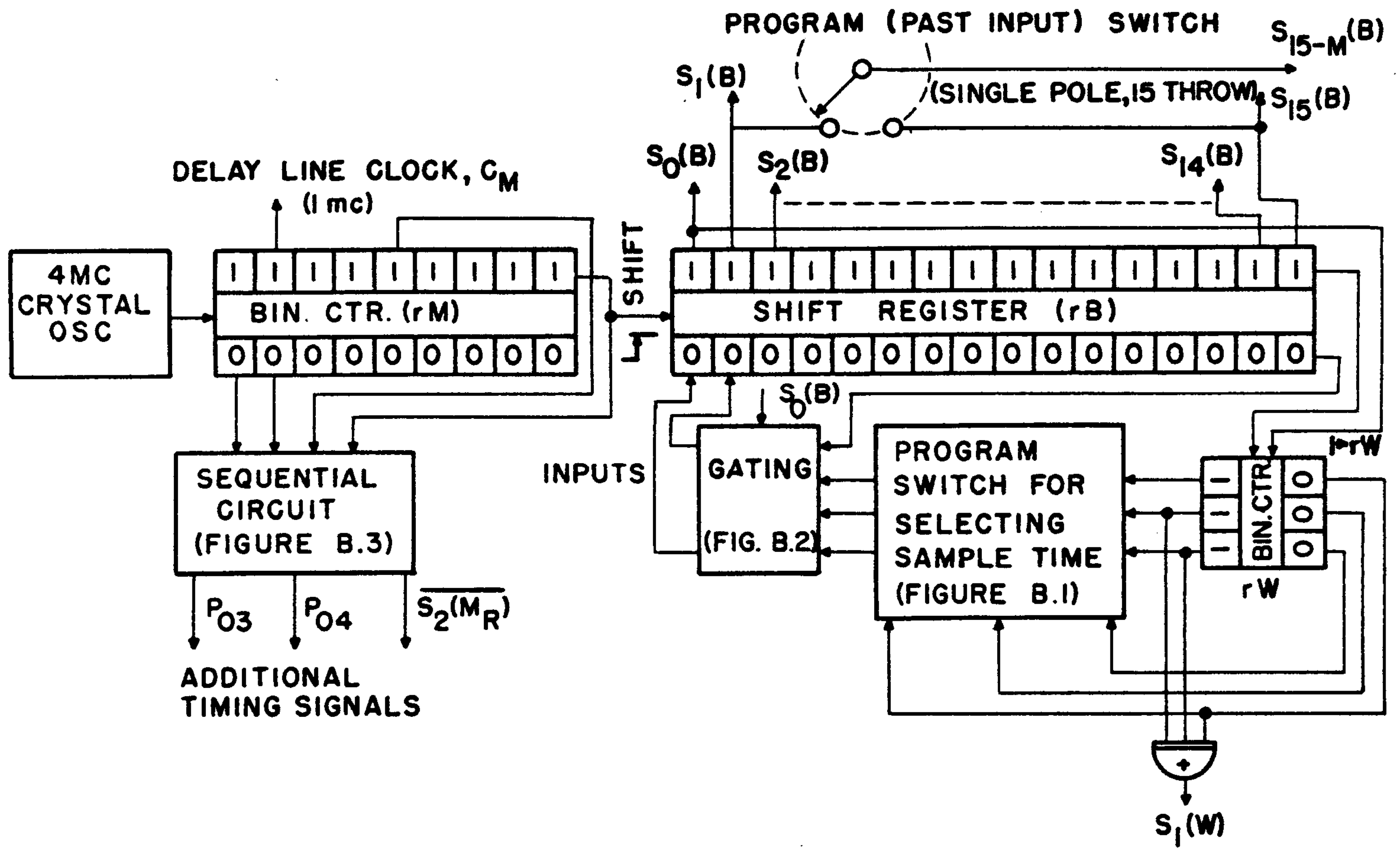


FIGURE 3.3 CONTROL LOGIC

synchronized with C_M was needed in the arithmetic unit. A four megacycle oscillator was therefore counted down to generate the one megacycle clock for the memory.

In designing the machine it was conceptually easier to consider C_M as the master clock since the delay line and most other circuits are clocked with C_M. We shall therefore call C_M the master clock throughout this report.

In keeping with this convention, it is convenient to define a register, rM_R, consisting of the seven stages of rM furthest to the right. rM is a nine bit binary counter which accumulates the four megacycle oscillator output. Therefore, rM_R is a seven bit binary counter which is triggered by C_M. Each overflow from this counter signifies that the next bit out of memory is the first bit of a new word. This overflow is used to change the state of rB. rB is a shift register with feedback to the first two stages. The function that is shifted into the first two stages is obtained from the feedback network while the remainder of the register operates as a normal shift register.

The operation of rB is very similar to the operation of a ring counter; only one of the output functions, $S_o(B), \ldots, S_{15}(B)$, equals "1" at any given time. The register is started with $S_o(B) = 1$ and this "1" is shifted along by succeeding overflows of rM_R. When the "1" is shifted out of the register, rW is incremented to remember the fact that one iteration of the computation cycle has been completed. Also, a "1" is shifted into either $C_o(B)$ or $C_1(B)$ depending upon the number of iterations that have been programmed. This "1" is then shifted along as before. Whenever rB leaves state $S_{15}(B)$, it is sent to $S_o(B)$ if the count in rW equals the number of iterations which have been programmed; otherwise it is sent to state $S_1(B)$. $S_o(B)$ is fed back to reset rW to a count of "1" whenever rB enters state $S_o(B)$.

A sixteen state shift register was used instead of a four bit counter with gating for two reasons:

1. It has been shown that this solution is more efficient in the total amount of logic required.[16]
2. It would be difficult to drive the number of gate legs required in the four bit counter solution.

The control signal $S_{15-M}(B)$ which, in conjunction with $S_1(W)$, defines the time at which the latest error is to be read into memory, is obtained by selecting the appropriate output function of rB with a rotary switch. $S_1(W)$ is "1" only when $rW = 1$ and therefore defines the time during which the first iteration of the computation cycle is being processed.

The mysterious sequential circuit appearing in Figure 3.3 supplies timing signals for other miscellaneous operations and is discussed in Appendix B.

3.7 Summary of the Compensator Design

The purpose of this section is to unify the foregoing presentation and to add a little more detail. The complete design is shown in Figure 3. 4.

Operation of the control logic should be clear. Note, however, that the complements of the control levels $S_1(B), \ldots, S_{15}(B)$ are required to gate out the constants K_i; the zero side of rB is therefore used.

The binary operational multiplier does not appear explicitly in Figure 3. 4; it is composed of rM and the AC coupled combinational circuit.

The combinational circuit which forms the sign of the product $K_i X_{n-i}$ is more complicated than that shown in Figure 3. 2 because this gating must select the proper bit in memory and must also select the sign of the latest input sample at the appropriate time.

It should not be assumed from Figure 3. 4 that the inputs to the various combinational circuits are always in their asserted form. The illustrated inputs merely serve to define the functional dependence of the circuits.

Some circuits have not been shown. These are the combinational circuits which define the operation of the compensator in the event it saturates and the circuits which generate the timing signals for the analog to digital converter and zero-order hold network.

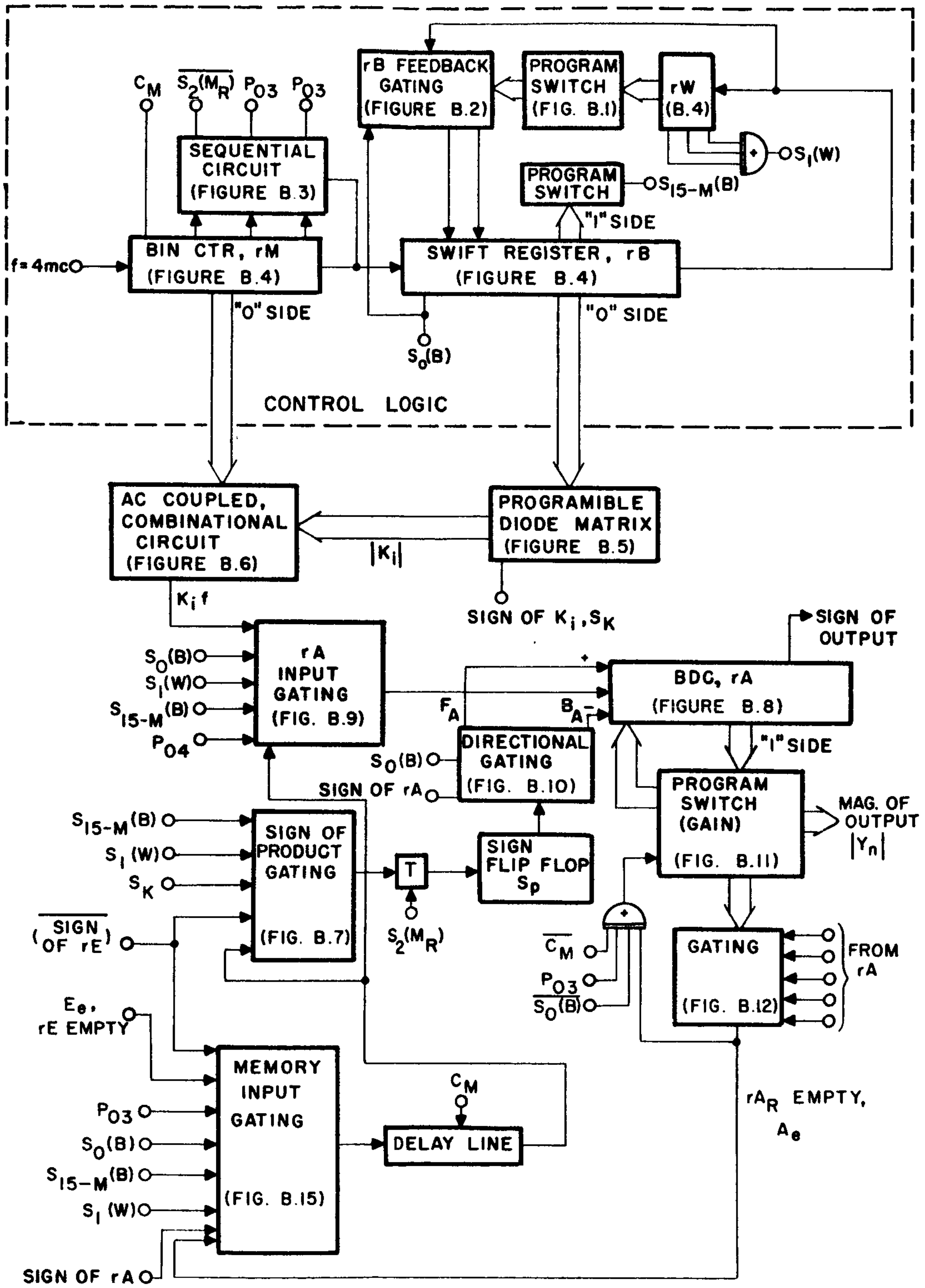


FIGURE 3.4– SIMPLIFIED COMPENSATOR LOGIC

A third (and final) program switch, used to set the gain, has been introduced. This switch has six possible settings. At setting number one, the first seven flip-flops of rA, $C_o(A), \ldots,$ $C_6(A)$ are used for the magnitude of the output; at setting number two, flip-flops $C_1(A), \ldots, C_7(A)$ are used, etc.

Since the output is not necessarily taken from the initial stages of rA, the gated clock which counts the output toward zero as the output is inserted into the delay line must enter the counter on a trigger input which bypasses the initial stages. The gain switch is also used to select the appropriate trigger input as shown schematically by the dashed line within the gain switch in Figure 3.4.

The feedback loop for writing the present output, y_n, into memory is now shown. This loop consists of a combinational circuit which informs the memory input when rA_R has been counted to zero by the master clock, C_M. rA_R is defined to consist of the cells of rA which form $|y_n|$ and all of the cells of rA to the right of these cells.

3.8 Programming the Compensator

It was shown in Section 2.1 that the compensator's transfer function can be written as

$$\frac{Y(z)}{E(z)} = D(z) = \frac{\sum_{i=0}^{N} \frac{a_i}{A} z^{-i}}{1 - \sum_{i=1}^{M} b_i z^{-i}} \quad . \qquad (2.5)$$

The various constants in this equation are defined in Section 2.1.

It is now convenient to write equation (2.5) in difference equation form:

$$y_n = \sum_{i=0}^{N} \frac{a_i}{A} e_{n-i} + \sum_{i=1}^{M} b_i y_{n-i} . \qquad (3.7)$$

The problem is to find the relation between the constants in equation (3.7) and the constants $\{K_i\}$ which are actually inserted into the diode matrix memory. Recall that these constants serve as inputs to the operational multiplier and must therefore have a magnitude smaller than or equal to one.

The diode matrix has fifteen rows, numbered one through fifteen, in which the constants $\{K_i\}$ are stored. The matrix was wired so that rows one through fifteen are gated in that order onto the input of the operational multiplier during each iteration of the computation cycle. If we now establish the convention that K_i will be stored in row i, then it should be clear from Figure 3.1, which defines the order that the stored samples emerge from memory, that the number

$$\sum_{i=0}^{N} 4K_{15-M-i}\, e_{n-i} + \sum_{i=1}^{M} 4K_{16-i}\, y_{n-i}$$

is accumulated in rA during one iteration of the basic computation cycle. The constants K_1 through $K_{15-M-N-1}$ are assumed to be zero. The factor 4 arises because the input frequency to the operational multiplier is four times that of the delay line clock. If p is the number of iterations of the basic computation cycle that have been programmed and the gain switch is at setting number one, then the first seven cells of rA define $|y_n|$ and we obtain:

$$y_n = \sum_{i=0}^{N} 4p\, K_{15-M-i}\, e_{n-i} + \sum_{i=1}^{M} 4p\, K_{16-i}\, y_{n-i}. \qquad (3.8)$$

If the gain setting is two, then the output of the operational multiplier is divided by two before it enters the output register. We therefore have

$$y_n = \sum_{i=0}^{N} \left(\frac{4p}{2}\right) K_{15-M-i}\, e_{n-i} + \sum_{i=1}^{M} \left(\frac{4p}{2}\right) K_{16-i}\, y_{n-i}. \qquad (3.9)$$

As the gain setting is increased, the output of the operational multiplier is further divided by powers of two before it enters the output register. We therefore obtain the relation

$$y_n = \sum_{i=0}^{N} \frac{4p}{2^{K-1}}\, K_{15-M-i}\, e_{n-i} + \sum_{i=1}^{M} \frac{4p}{2^{K-1}}\, K_{16-i}\, y_{n-i}. \qquad (3.10)$$

K is the gain setting and may assume values one through six. If we make the definition

$$K_D = 2^{3-K}\, p \quad , \qquad (3.11)$$

then equation (3.10) becomes:

$$y_n = \sum_{i=0}^{N} K_D K_{15-M-i}\, e_{n-i} + \sum_{i=1}^{M} K_D K_{16-i}\, y_{n-i}\,. \qquad (3.12)$$

We shall call K_D the digital gain. Comparison of equations (3.7) and (3.12) yields the expressions for the constants $\{K_i\}$:

$$K_{15-M-i} = \frac{a_i}{AK_D}\,, \; i = 0, 1, \ldots, N, \qquad (3.13)$$

and

$$K_{16-j} = \frac{b_j}{K_D}\,, \; j = 0, 1, \ldots, M. \qquad (3.14)$$

The remaining constants, of course, are zero. The digital gain, K_D, is first selected so that

$$\frac{|b_j|}{K_D} \leq 1, \text{ for all } j \qquad (3.15)$$

and then the analog gain is selected so that

$$\frac{|a_i|}{AK_D} \leq 1, \text{ for all } i. \qquad (3.16)$$

The constants $\{K_i\}$ are then inserted into the proper rows of the diode matrix. Unless the constant equals ± 1, its magnitude is encoded in seven bit natural binary with positive numbers denoted by binary "zero" in the sign bit. The numbers are then inserted from left to right with the sign bit in column A, the most significant bit in column B, and the least significant bit in column H. The presence of a diode corresponds to binary "one". If the constant is ± 1, the sign is programmed as above, but columns B through H are left open and a diode is inserted into column J. This bypasses the operational multiplier and gates its input frequency directly into rA.

The maximum digital gain, K_{DMAX}. is an important number because it is equal to the maximum constant, b_j, which can be programmed. K_{DMAX} is a function of p:

$$K_{DMAX} = 2^{(3-1)}p = 4p. \tag{3.17}$$

Therefore, a wider range of constants may be programmed when larger sample times are used.

The programming procedure may now be summarized:

1. Select a sample time commensurate with the dominant time constants of the analog dynamics.
2. Synthesize K(z).
3. Select the largest gain setting (smallest K_D) such that $|b_j|/K_D \leq 1$, for $j = 1, \ldots, M$.
4. Select the smallest analog gain such that $|a_i|/AK_D \leq 1$ for $i = 0, \ldots, N$.
5. Insert the constants b_j/K_D and a_i/AK_D into the diode matrix.
6. Set the past input switch to position 15-M.
7. Set the sample time switch. The numbers on the dial plate correspond to the sample time rounded off to the nearest even integer.

The smallest digital gain possible should be used since this will result in the maximum "smoothing" of the operational multiplier output frequency and hence will yield the highest computational accuracy. This also enables the constants which contribute significantly to the output to be encoded with minimum truncation error, since these constants will then lie close to one. Any real number with a magnitude between one and 100/128, for example, can always be represented by a number which is accurate to within .05 per cent. Real numbers with a magnitude smaller than 10/128, however, may have to be encoded with errors in excess of 5 per cent.

The smallest analog gain which satisfies equation (3.16) should be used also to ensure that the input constants α_i/AK_D are not too small for accurate encoding. A small analog gain is also desirable, of course, to minimize instrumentation and stability problems in a practical control loop.

Another inaccuracy which results from programming small constants has not yet been mentioned. When a constant is multiplied by a small number from memory, the contributions of the latter stages of rM to the output frequency of the operational multiplier will not be obtained. This happens because the output of the operational multiplier is only monitored for a small time if the number in memory is small. The resultant error is small if the dominant weighting constants are programmed as numbers close to one. If even the largest constants are programmed as small numbers, however, the compensator input would have to become quite large before an output from the multiplier would be obtained, resulting in a large dead zone.

CHAPTER FOUR
SYNTHESIS TECHNIQUES

4.1 Introduction

The main purpose of this chapter is to present those results from sampled-data theory which are directly applicable to the problem at hand. These results are stated, with condensed developments given where results in slightly different form from that stated in the literature were desired. Unless explicitly stated otherwise, the simplifying assumption is made throughout this chapter that the system is sampled but not quantized.

Although the mathematics involved in sampled-data theory is very straightforward, it does tend to become lengthy, as do the resultant equations. The intuitive grasp of system behavior, essential in fixing the many degrees of freedom in order to effect the proper trade-offs required to obtain a satisfactory response, is easily lost. Another important function of this chapter is to develop an intuitive grasp of the synthesis problem.

4.2 Restrictions upon the Selection of the Open Loop Pulse Transfer Function[21, 24]

The linear program for the digital compensator is obtained by inserting the desired open loop pulse transfer function, K(z), into equation (2. 3). The selection of K(z) is not completely arbitrary, but is restricted by the analog dynamics. There are three restrictions:

1. The lowest power of z^{-1} in the numerator of K(z) must be larger than or equal to the lowest power of z^{-1} in the numerator of G(z).
2. K(z) must have as its zeros all of the zeros of G(z) which lie on or outside the unit circle in the z-plane.
3. 1-K(z) must have as its zeros all of the poles of G(z) which lie on or outside the unit circle in the z-plane.

The first restriction assures us that the compensator's transfer function is physically realizable. When this constraint is satisfied, the compensator need not deliver an output before it receives an input. When the lowest power of z^{-1} in the numerator of K(z) is taken to be equal to the lowest power of z^{-1} in the numerator of G(z), the most recent error sample is utilized by the compensator. Since this minimizes the delay through the forward path of the loop, we shall always make this selection.

The last two restrictions assure us that the compensator is not called upon to cancel poles and zeros of G(z) which lie on or outside the unit circle. Imperfect cancellation of these zeros and poles will usually lead to closed loop instability.

If the analog plant is stable, as is always the case with practical actuators, the last restriction can be stated more exactly. In this case, G(z) will not have poles outside or on the unit circle, with one exception. It is very common to have a single or multiple pole at z=1, since a pole at z=1 corresponds to integration. We can therefore replace the last restriction with the following statement:

> If $G(z)$ represents a stable system and has a pole of order i at z=1, then $1-K(z)$ must be of the form $(1-z^{-1})^i P(z^{-1})$ where $P(z^{-1})$ is a polynomial in z^{-1}.

4.3 No Ripple Response[21, 24]

Usually a zero-order hold network is used in a digital system for ease of instrumentation and to obtain high null stability. If a zero-order hold network is used, then it is clear that a ripple free response to a polynomial input is possible only if the analog plant is capable of generating the required polynomial with a constant input (the output of the zero-order hold). For example, in order to generate a smooth ramp, an integration must be present in the analog plant.

Besides the restriction on the analog plant, $K(z)$ must be restricted so that its zeros include all of the zeros of $G(z)$. Of course, the three restrictions upon $K(z)$ which were listed in the previous section must also be satisfied. Note, however, that the second restriction is automatically satisfied when all of the zeros of $G(z)$ are also zeros of $K(z)$.

In the above discussion, the problem has been simplified by assuming that the zero-order hold network is capable of supplying any real constant output. In a digital control loop, however, the output of the hold network is restricted to a set of discrete values. Unless one of these values is the input that is required to generate the required polynomial to within the quantization error, we can expect to see at least one quantum ripple on the output even though the above constraints have been satisfied.

4.4 Constraint Equations for Smoothing and Prediction with Finite Settling Time[21]

The three basic operations of prediction, smoothing, and differentiation can be performed with a linear computer program of the following form:

$$c_n = \sum_{i=0}^{J} \gamma_i \, r_{n-j-i} \quad , \tag{4.1}$$

where c(t) is the output and r(t) is the input. J, n, and j are non-negative integers. This formula is called a rough history, or finite memory program, because only a fixed number of past input samples need be remembered in order to generate the output.

If the input is a polynomial for $t \geq 0$, the constants $\{\gamma_i\}$ can be selected so that for any real number, β, the equation

$$c_n = r_{n+\beta} \tag{4.2}$$

is satisfied for every n larger than or equal to N + j, regardless of the input prior to time zero or the initial conditions. It is in this sense that we speak of a finite settling time. If β is larger than zero, the program predicts the input.

It is possible to select the constants $\{\gamma_i\}$ so that the input is smoothed for any value of β while equation (4.2) is simultaneously satisfied. This process is described in the next section. The constants may also be selected so that the output equals the derivative of the input in the same sense as described above (p. 127, ref. 21), but we shall have no need for this function. Let us therefore return to the problem of selecting the constants $\{\gamma_i\}$ so that equation (4.2) is satisfied. We begin by combining equations (4.2) and (4.1):

$$r_{n+\beta} = \sum_{i=0}^{J} \gamma_i \, r_{n-j-i} \quad . \qquad (4.3)$$

If r(t) is a polynomial of degree q, a Taylor series expansion of r(t) about the point $t = nT - jT$ yields:

$$r(t) = r(nT - jT) + \sum_{m=1}^{q} \frac{(t - nT + jT)^m}{m!} \frac{d^m r}{dt^m} (nT - jT) . \qquad (4.4)$$

Of course, the summation in equation (4.4) is absent if q is zero. Equation (4.4) implies that

$$r_{n+\beta} = r(nT - jT) + \sum_{m=1}^{q} \frac{(\beta T + jT)^m}{m!} \frac{d^m r}{dt^m} (nT - jT) , \qquad (4.5)$$

and

$$r_{n-j-i} = r(nT - jT) + \sum_{m=1}^{q} \frac{(-iT)^m}{m!} \frac{d^m r}{dt^m} (nT - jT) \quad . \qquad (4.6)$$

Combining equations (4.3), (4.5), and (4.6), we obtain:

$$r(nT-jT) + \sum_{m=1}^{q} \frac{(\beta T + jT)^m}{m!} \frac{d^m r}{dt^m} (nT - jT) =$$

$$\sum_{i=0}^{J} \gamma_i \left[r(nT - jT) + \sum_{m=1}^{q} \frac{(-iT)^m}{m!} \frac{d^m r}{dt^m} (nT - jT) \right], \tag{4.7}$$

or

$$\left[\left(\sum_{i=0}^{J} \gamma_i \right) - 1 \right] r(nT - jT) +$$

$$\sum_{m=1}^{q} \left[\left(\sum_{i=1}^{J} \frac{\gamma_i (-iT)^m}{m!} \right) - \frac{(\beta T + jT)^m}{m!} \right] \frac{d^m r}{dt^m} (nT - jT) = 0. \tag{4.8}$$

If this equation is to hold for every polynomial of degree q, the quantities in brackets must be identically zero:

$$\left[\sum_{i=0}^{J} \gamma_i \right] - 1 = 0,$$

$$\left[\sum_{i=1}^{J} \frac{\gamma_i (-iT)^m}{m!} \right] - \frac{(\beta T + jT)^m}{m!} = 0, \tag{4.9}$$

$$m = 1, 2, \ldots, q.$$

We therefore obtain the following set of equations:

$$\sum_{i=0}^{J} \gamma_i = 1$$

$$\sum_{i=0}^{J} i \gamma_i = -(\beta + j), \qquad m = 1 \tag{4.10}$$

$$\vdots$$

$$\sum_{i=0}^{J} i^q \gamma_i = (-1)^q (\beta + j)^q, \qquad m = q.$$

Therefore, if we wish to satisfy equation (4.2) for an input polynomial of degree q, the constants $\{\gamma_i\}$ in equation (4.1) must satisfy the q + 1 equations in equation set (4.10). Satisfaction of constraints (4.10) is also a sufficient condition to ensure the validity of equation (4.2) for polynomial inputs, since we can begin with equation set (4.10) and perform, in reverse order the steps that were followed above.

Since we must have at least as many unknowns as equations, it follows that we must have

$$J \geq q. \tag{4.11}$$

If J equals q, the constants are uniquely determined by solving equation set (4.10). When J is chosen larger than q, the extra degrees of freedom can be fixed so that equation (4.1) also performs the function of smoothing.

If equation (4.1) represents a closed loop transfer function, J must usually be chosen larger than q, where q is the highest order polynomial that the closed loop must follow. It is then possible to satisfy equation set (4.10) and use the extra degrees of freedom to satisfy the additional constraints imposed upon the $\{\gamma_i\}$ by the restrictions discussed in Sections 4.2 and 4.3. Once all of these constraints are satisfied, we are free to impose any additional constraints until the number of constants, J + 1, equals the total number of constraint equations. These additional constraints may, for example, be selected to minimize noise or to obtain some desired transient response.

Since equation (4. 1) yields a finite settling time, the closed loop transfer function, K(z), is usually taken to be of the form obtained by z- transforming equation (4. 1):

$$\frac{G(z)}{R(z)} = K(z) = z^{-j} \sum_{i=0}^{J} \gamma_i z^{-i} . \qquad (4.12)$$

We therefore see, from considerations presented in Section 4. 2, that in this particular application j will equal the lowest power of z^{-1} in the numerator of G(z). This is the reason that the parameter j was not set equal to zero in equation (4. 1). If the selection of j was arbitrary, it would always be chosen to be zero, the smallest possible value which yields a physically realizable equation. A positive j serves no useful purpose and only increases the time delay through the filter.

The case $\beta = 0$ is of prime interest since this corresponds to perfect reproduction of the input signal at the sampling instants. For this case, equation set (4. 10) reduces to

$$\sum_{i=0}^{J} \gamma_i = 1,$$

$$\sum_{i=0}^{J} i^m \gamma_i = (-1)^m j^m, \qquad (4.13)$$

$$m = 1, 2, \ldots, q.$$

4.5 Noise Reduction with Finite Settling Time Filters[21]

The reduction of noise is important even in applications where the input is not contaminated with noise. W. R. Bennett[3] has shown that under certain conditions, quantizers can be replaced by white noise sources. Even when these conditions are not satisfied, intuition strongly suggests that a control loop which is designed to reject noise will also reduce the severity of any ripple caused by quantization.

Another advantage of designing for minimum noise is that sudden changes in the closed loop input are smoothed in an optimal manner. For example, as the noise rejection of a closed loop is increased, the amount of overshoot that is present in the response to a step input becomes smaller. (See Sections 4.7 and 5.2.) Intuitively, this occurs because when a system which is designed to reject noise receives a step input, it does not know whether it has actually received a step input and should respond, or whether it has received a noise spike and should remain stationary. The filter must therefore respond conservatively. The input is approached slowly and the tendency to overshoot is reduced.

It has been shown (Chapter 9, ref. 21) that noise reduction can be accomplished when a finite memory linear program (equation 4.1) is used by minimizing the function $\sum_{i=0}^{J} \gamma_i^2$, which is called the variance reduction factor. It makes no sense to minimize this function by itself since the obvious minimum of zero is obtained for $\gamma_i = 0$ for every i. Not only would this program reject all noise, but all signals as well! Constraints must be imposed upon the constants to assure that the desired signals are passed.

We shall concentrate on minimizing the variance reduction factor subject to the constraints that the filter must respond to step and ramp inputs with small error as is usually the case when the filter is a closed loop control system. These constraints imply that equation set (4. 10) must be satisfied for q = 1, i. e., the first two equations in the set must be satisfied:

$$\sum_{i=0}^{J} \gamma_i = 1 , \qquad (4.14a)$$

$$\sum_{i=0}^{J} i\,\gamma_i = -(\beta + j) . \qquad (4.14b)$$

Even though the closed loop will respond to a ramp input with no error when β is zero, it is not advisable to set β equal to zero at this time. Since it is possible to improve the step response and noise rejection of the closed loop by accepting a time lag in the response to a ramp, a negative β may represent the best compromise in some applications. Also, the interesting possibility of employing a positive β in order to compensate for a known output load exists.

When the variance reduction factor is minimized for constant β, j, and J, subject to the above constraints, (pp. 118-123, ref. 21), we obtain

$$\left[\sum_{i=0}^{J} \gamma_i^2\right]_{\min} = \frac{2J(2J+1) + 12\alpha(J+\alpha)}{J(J+1)(J+2)}, \qquad (4.15)$$

$$\alpha = \beta + j$$

The constants which minimize the variance reduction factor are given by:

$$\gamma_i = \frac{2(2J+1) + 6\alpha}{(J+1)(J+2)} - \frac{6(J+2\alpha)}{J(J+1)(J+2)}\, i. \qquad (4.16)$$

It was stated above that the noise rejection of a system can be increased if (for constant J) the time lag in response to a ramp is increased. This can be shown by finding the α which yields the maximum noise reduction for a given J. This α is found by differentiating equation (4. 15) with respect to α and setting the result equal to zero. α is found to be $-J/2$. Therefore:

$$\beta = -(j + J/2). \qquad (4.17)$$

Since, according to equation (4. 15), noise sensitivity increases monotonically as β is increased from the value given by equation (4. 17), it follows that β should be taken as close to the negative number $-(j + J/2)$ as possible for maximum noise rejection. This increases the time lag through the system.

In most cases, equation (4.16) cannot be used to define a closed loop transfer function, because K(z) must also be constrained to satisfy all of the restrictions described in Section 4.2. The first restriction has already been accounted for by including the parameter j in the above equations. Also, if we assume that G(z) does not have a pole of higher order than two at $z = 1$ (as is usually the case), the third restriction has been automatically satisfied by imposing constraints (4.14). (If the pole at $z = 1$ is of order two, then β must be taken to be zero. If there is no pole at $z = 1$ or if the pole is of order one, then β may assume any real value.) It is the second restriction which forces a modification of the above development. Whenever K(z) is to have as its zeros some of the zeros of G(z), either because they lie outside the unit circle or because a ripple-free response is desired, additional constraints must be imposed.

Let us assume that K(z) is to have as its zeros two of the zeros of G(z). Then K(z) is of the following form:

$$K(z) = z^{-j} \sum_{i=0}^{J} \gamma_i z^{-i} = (1 - z_1 z^{-1})(1 - z_2 z^{-1}) z^{-j} \sum_{i=0}^{J-2} \beta_i z^{-i}, \tag{4.18}$$

where z_1 and z_2 are the zeros of G(z). If we let $u_1 = -z_1 - z_2$ and $u_2 = z_1 z_2$, equation (4.18) implies that

$$\sum_{i=0}^{J} \gamma_i z^{-i} = (1 + u_1 z^{-1} + u_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i}. \tag{4.19}$$

Now, by equating coefficients of like powers of z^{-1} on each side of this equation, we obtain expressions for the constants $\{\gamma_i\}$ in terms of the constants $\{\beta_i\}$. This enables us to write the variance reduction factor and equations (4.14) in terms of the constants $\{\beta_i\}$. The variance reduction factor becomes

$$\sum_{i=0}^{J} \gamma_i^2 = \beta_o^2 + (\beta_1 + u_1\beta_o)^2 + \sum_{i=2}^{J-2} (\beta_i + u_1\beta_{i-1} + u_2\beta_{i-2})^2$$
$$+ (u_1\beta_{J-2} + u_2\beta_{J-3})^2 + (u_2\beta_{J-2})^2 , \tag{4.20}$$

and the two constraint equations become

$$\sum_{i=0}^{J-2} \beta_i = 1/(1 + u_1 + u_2) , \tag{4.21a}$$

$$\sum_{i=0}^{J-2} i\beta_i = \delta , \tag{4.21b}$$

where
$$\delta = - \frac{\alpha + u_1(\alpha + 1) + u_2(\alpha + 2)}{(1 + u_1 + u_2)^2} . \tag{4.22}$$

The variance reduction factor is now minimized subject to constraints (4.21) by treating the β_i's as independent variables. This leads to the set of linear algebraic equations which are written in matrix form on the next page. The variables λ_1 and λ_2 are Lagrange multipliers which are introduced for convenience. These equations define the constants $\{\beta_i\}$ which yield the smallest variance reduction factor subject to constraints (4.14) and subject

$$\begin{bmatrix}
1 & 0 & (1+u_1^2+u_2^2) & (u_1+u_1u_2) & u_2 & 0 & \cdots & \cdots & 0 \\
1 & 1 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) & (u_1+u_1u_2) & u_2 & 0 & \cdots & 0 \\
1 & 2 & u_2 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) & (u_1+u_1u_2) & u_2 & 0 \cdots & 0 \\
1 & 3 & 0 & u_2 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) & (u_1+u_1u_2) & u_2 \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
1 & (J-4) & 0 \cdots & 0 & u_2 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) & (u_1+u_1u_2) & u_2 \\
1 & (J-3) & 0 & \cdots & 0 & u_2 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) & (u_1+u_1u_2) \\
1 & (J-2) & 0 & \cdots & \cdots & 0 & u_2 & (u_1+u_1u_2) & (1+u_1^2+u_2^2) \\
0 & 0 & 1 & 1 & \cdots & \cdots & \cdots & \cdots & 1 \\
0 & 0 & 0 & 1 & 2 & \cdots & \cdots & \cdots & J-2
\end{bmatrix}
\begin{bmatrix}
\lambda_1 \\ \lambda_2 \\ \beta_0 \\ \beta_1 \\ \vdots \\ \vdots \\ \beta_{J-4} \\ \beta_{J-3} \\ \beta_{J-2}
\end{bmatrix}
=
\begin{bmatrix}
0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ \dfrac{1}{1+u_1+u_2} \\ \delta
\end{bmatrix}
\qquad (4.23)$$

$$\begin{bmatrix} 1 & 0 & (1+u_1^2) & u_1 & 0 & 0 & 0 \cdots & 0 \\ 1 & 1 & u_1 & (1+u_1^2) & u_1 & 0 & 0 \cdots & 0 \\ 1 & 2 & 0 & u_1 & (1+u_1^2) & u_1 & 0 \cdots & 0 \\ 1 & 3 & 0 & 0 & u_1 & (1+u_1^2) & u_1 \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (J-2) & 0 \cdots & \cdots & 0 & u_1 & (1+u_1^2) & u_1 \\ 1 & (J-1) & 0 \cdots & \cdots & \cdots & 0 & u_1 & (1+u_1^2) \\ 0 & 0 & 1 & 1 & \cdots & \cdots & \cdots & 1 \\ 0 & 0 & 1 & 2 & 3 & \cdots & \cdots & (J-1) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \beta_0 \\ \beta_1 \\ \vdots \\ \vdots \\ \vdots \\ \beta_{J-2} \\ \beta_{J-1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ \frac{1}{1+u_1} \\ -\frac{a+u_1(a+1)}{(1+u_1)^2} \end{bmatrix} \qquad (4.24)$$

to the constraint that K(z) have as its zeros the two zeros of G(z). Once the constants $\{\beta_i\}$ are found, K(z) is determined from equation (4.18). Although the solution of the matrix equation (4.23) may be quite tedious when performed manually, this synthesis procedure is simple and rapid if one has access to a general purpose digital machine.

The details of the above development may be found on pages 234 to 237 of reference 21 for the simpler case of constraining K(z) to have only one known zero. In this case the equations for the constants $\{\beta_i\}$ are defined by matrix equation (4.24), and K(z) is given by:

$$K(z) = (1 + u_1 z^{-1})\, z^{-j} \sum_{i=0}^{J-1} \beta_i\, z^{-i}\,, \tag{4.25}$$

where $u_1 = -z_1$, the known zero. This result can also be obtained as a special case of the two-zero result by letting $z_2 = 0$ and realizing that the summations must be performed with the upper limit J - 1.

If G(z) has a pole of higher order than two at z = 1, the third restriction in Section 4.2 is best-handled by designing K(z) to pass a polynomial of sufficiently high order. Then this restriction is satisfied and the closed loop following capability is simultaneously improved. If the closed loop is to follow a second degree polynomial, the following constraint must be imposed upon K(z) in addition to constraints (4.14):

$$\sum_{i=0}^{J} i^2 \gamma_i = a^2 . \tag{4.26}$$

When K(z) is restricted to have two known zeros z_1 and z_2, K(z) is given by equation (4.18) and the constants $\{\beta_i\}$, which yield the minimum variance reduction factor, are defined by a matrix equation obtained by the same minimization procedure as before. It should be clear that this procedure can also be used when K(z) must have more than two known zeros, or when K(z) must pass higher order polynomials, but the algebra becomes formidable.

4.6 Finite Settling Time Step Response

The application of constraints (4.10) provides an excellent method of limiting the error that is present in a control loop while it is operated within its bandwidth. However, the transient response of a control loop is usually of prime importance, either because the control problem demands that the control system respond well to sudden changes in the input, or because the control loop must be subjected to disturbances. Emphasis is placed upon the step response since it can be easily measured and provides an excellent indication of the closed loop dynamics.

One synthesis procedure has been developed which incorporates transient considerations (Chapter 12, ref. 21), but which cannot be applied if the closed loop transfer function must incorporate some known zeros. The purpose of this section is to introduce another synthesis procedure which can be applied when K(z) is to have known zeros. The technique enables one to select the constants in the closed loop transfer function so that the step response assumes prescribed values at certain sampling instants during the transient response. In those cases where K(z) need not incorporate any known zeros and where the constants need only satisfy equation (4. 14a), this method enables one to synthesize K(z) to yield any desired step response that is physically realizable. If i constraints must be imposed upon K(z) in addition to constraint (4. 14a), then only J - i points on the transient portion of the step response can be specified. For example, if constraint (4. 14b) must also be satisfied and K(z) must have two known zeros, then only J - 3 points on the transient portion of the step response can be specified. The remaining points are defined by the constraints. In these cases, the specified points must be selected by trial and error until the unspecified and specified points result in an acceptable transient trajectory. Although this trial and error procedure may appear to be objectionable, the procedure can be efficiently applied in many cases (see Section 5. 3), particularly if a general purpose digital machine is available. It must be remembered that every sampled-data synthesis procedure is in fact a trial and error procedure when transient considerations are important and K(z) must satisfy numerous constraints. One must synthesize, examine the results, and repeat until a suitable transient response is obtained.

The development of the synthesis procedure follows. A unit step input is defined by

$$r_n = \begin{cases} 1, & n = 0,\ 1,\ 2,\ \ldots \\ 0, & n = -1,\ -2,\ \ldots \quad . \end{cases} \tag{4.27}$$

When this equation is inserted into equation (4. 1), we obtain:

$$c_n = \begin{cases} 0 & , \quad n < j \\ \sum_{i=0}^{n-j} \gamma_i & , \quad n = j,\ j+1,\ \ldots,\ j+J-1 \\ \sum_{i=0}^{J} \gamma_i & , \quad n = j+J,\ j+J+1,\ \ldots \quad . \end{cases} \tag{4.28}$$

If the steady state output is to equal the input, the same equation that was obtained in Section 4. 3 with the more general treatment results: $\sum_{i=0}^{J} \gamma_i = 1$. The important fact is that $c_n = \sum_{i=0}^{n-j} \gamma_i$ when $n = j,\ j+1,\ \ldots,\ j+J$. Therefore, when K(z) need satisfy no constraints other than that imposed by the step response, we can select the constants $\{\gamma_i\}$ so that c_n assumes any value we desire during the transient period. More explicitly, we can write:

$$c_j = \gamma_o$$

$$c_n - c_{n-1} = \sum_{i=0}^{n-j} \gamma_i - \sum_{i=0}^{n-1-j} \gamma_i\ , \quad n = j+1,\ j+2,\ \ldots,\ j+J, \tag{4.29}$$

or

$$c_n - c_{n-1} = \gamma_{n-j} \tag{4.30}$$

for n = j, j + 1, . . . , j + J. If we let i = n - j, the equations for the constants become

$$\gamma_i = c_{i+j} - c_{i+j-1}, \quad i = 0, 1, \ldots, J. \tag{4.31}$$

Figure 4.1 clearly illustrates this simple result. We need only specify the outputs we desire at the sampling instants and insert them into equation (4.31) in order to obtain the constants $\{\gamma_i\}$. Noise rejection is simultaneously accomplished by making the magnitude of the difference between successive outputs small, since this results in small γ_i^2's. As stated earlier, however, all of the transient outputs cannot be specified arbitrarily when constraints other than equation (4.14a) must be satisfied.

The constraint equations that the constants $\{\gamma_i\}$ must satisfy if K(z) is to have k known zeros are obtained by dividing the factor which defines all of the known zeros, $(1 + u_1 z^{-1} + \ldots + u_k z^{-k})$, into K(z) and setting the coefficients of z^{-1} in the numerator of the remainder equal to zero. These equations, together with equations (4.10), establish k + q + 1 equations for the constants $\{\gamma_i\}$. $J + 1 - (k + q + 1) = J - (k + q)$ equations in set (4.31) may therefore be used to constrain the transient response. An example of this procedure is given in Section 5.2 .

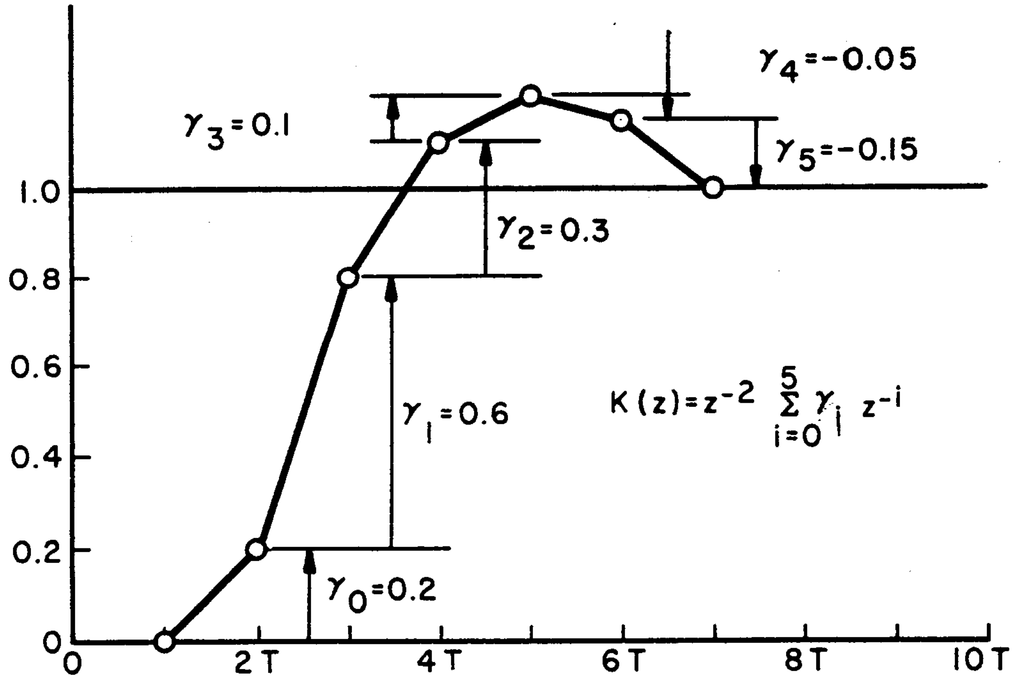


FIGURE 4.1- RELATION BETWEEN K(z) AND THE UNIT STEP RESPONSE

4.7 Transient Considerations

It was mentioned in Section 4.5 that as the noise rejection of a closed loop is increased, the amount of overshoot that is present in the step response diminishes. We are now in a position to better understand this statement. If the steady state error for a ramp input is specified, then the constants $\{\gamma_i\}$ must satisfy constraints (4.14):

$$\gamma_o + \gamma_1 + \dots + \gamma_J = 1 \tag{4.14a}$$

$$\gamma_1 + 2\gamma_2 + \dots + J\gamma_J = -(\beta + j). \tag{4.14b}$$

The ideal step response should increase monotonically to the final value. Since equation (4.31) is always valid, regardless of how the constants $\{\gamma_i\}$ are selected, all of the constants $\{\gamma_i\}$ must be positive in order to obtain the ideal step response. However, if the closed loop is to also follow a ramp with no steady state error, equation (4.14b) implies that $\sum_{i=0}^{N} i\gamma_i = -j$. This implies that some of the constants $\{\gamma_i\}$ must be negative. Furthermore, since the γ_i's are weighted more heavily for large i in equation (4.14b), it follows that the step response must overshoot. It is therefore impossible to design a linear sampled-data system with a finite settling time which will have zero steady state error with ramp and step inputs and which will have no overshoot with a step input. As β is made more negative, the magnitude of the negative γ_i's can be smaller so that step responses with less overshoot are possible.

When β is less than or equal to - j, step responses with no overshoot are possible. Since β must also be reduced in order to increase noise rejection (see Section 4. 5), it follows that designing for increased noise rejection will also reduce the overshoot present in the step response.

We have just shown that the transient step response of a linear, sampled-data system can be improved by sacrificing steady state ramp accuracy. In general, whenever increased steady state performance is demanded of a linear, sampled-data system, its transient response must suffer. Let us assume, for example, that the control system must follow a second degree polynomial. The following equation must then be satisfied in addition to constraints (4. 14):

$$\sum_{i=0}^{J} i^2 \gamma_i = (\beta + j)^2 \tag{4.32}$$

Again, let us assume that β is zero so that the steady state error is zero. In the above, we concluded that in order to satisfy constraints (4. 14) the constants $\{\gamma_i\}$ should be negative for large i. This, however, would cause the left hand side of equation (4.32) to be negative. If all three constraints are to be satisfied and the γ_i's to be kept small in order to keep the variance reduction factor small, the constants which assume negative values must be those corresponding to intermediate values of i. When this statement is interpreted in terms of Figure 4. 1, it follows that the step response will exhibit an oscillation. The magnitude of the constants $\{\gamma_i\}$ will also increase and result in increased overshoot.

It is tempting to indiscriminately employ constraint equations (4. 10) with a large q since it would appear that the following capability of the control system could be greatly improved without much increase in the memory capacity of the compensator. It must always be remembered, however, that these equations only define the steady-state behavior of the control system and say nothing about its transient behavior. As q is increased, the oscillating sign on the right hand side of constraint equations (4. 10) results in oscillations of increased amplitude and frequency in the time domain.

CHAPTER FIVE
EXPERIMENTAL EVALUATION OF THE COMPENSATOR

5.1 Experimental Control Loop

The experimental control loop used to evaluate the compensator is shown in Figure 5.1. A photograph of the equipment is shown in Figure 5.2. A small analog computer (Appendix C) was used for the analog plant so that the compensator could be tested with various analog dynamics.

Although the error was generated in analog form, it should not be implied that the use of this compensator in analog control loops is proposed. This instrumentation was merely selected as a convenient method of simulating the digital control loop shown in Figure 1.3. With this instrumentation, a digital subtractor was not required, and test inputs could be conveniently obtained from an analog signal generator. Although some uncompensated, analog closed loop responses are shown in this chapter, they are shown only to illustrate the best response that could possibly be obtained with an uncompensated, digital control loop with identical gain and analog plant.

The analog to digital converter establishes a quantum size of fifty millivolts per quantum. The maximum error the A/D converter can accept before it saturates is $\pm$ 127 quanta, or

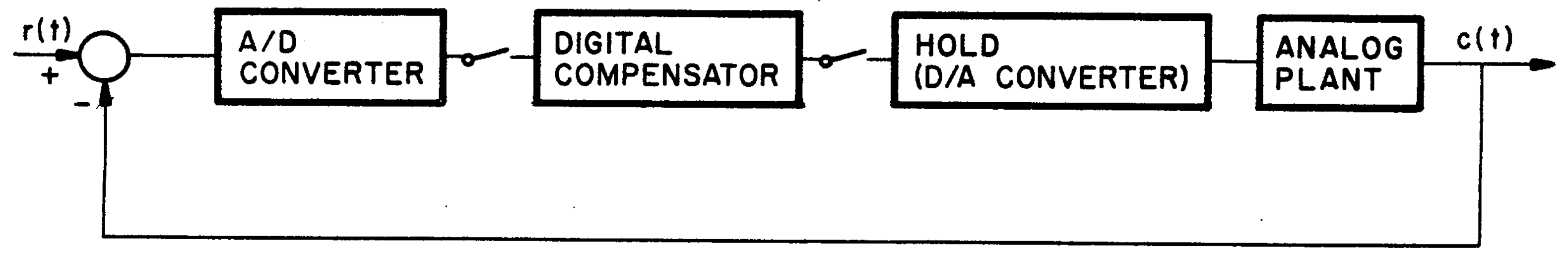


FIGURE 5.1— EXPERIMENTAL CONTROL LOOP

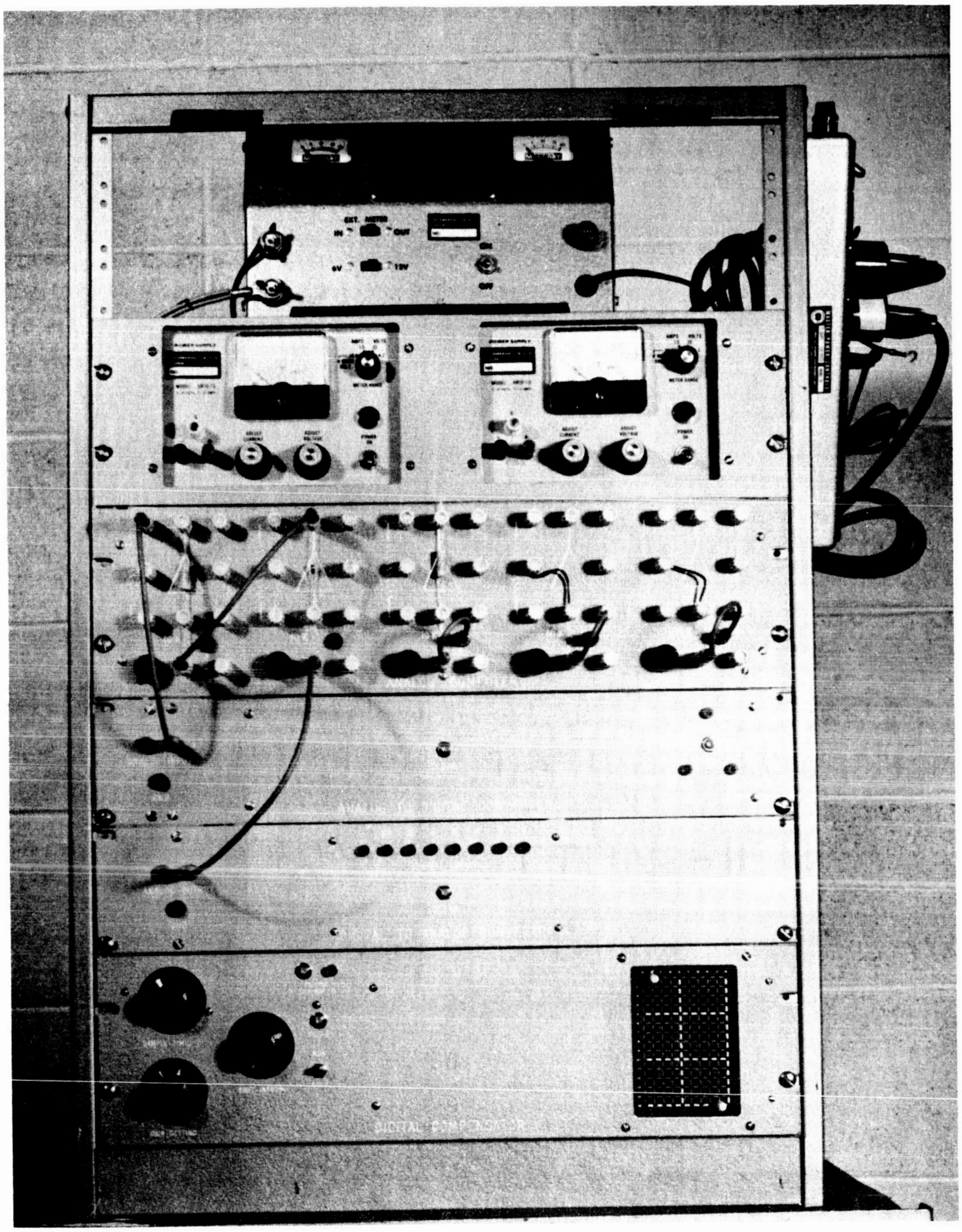


FIGURE 5.2 – PHOTOGRAPH OF THE EXPERIMENTAL CONTROL LOOP

$\pm$ 6.35 volts. The fifty millivolt quantum size is sufficiently large for analog drift and noise to be negligible, and small enough to result in a maximum signal which can be easily handled with transistor circuits.

The fact that analog feedback was used in the test loop does not imply that the test loop does not simulate high precision digital feedback systems. Although fifty millivolts represents a relatively coarse quantization of the output variable in the test loop, which saturates at about $\pm$ 10 volts, this output range may represent only a small portion of the full range of the output variable in an actual digital control loop. Therefore, except for the extra error introduced by quantizing the input signal for a strictly digital control loop, the test results accurately define the behavior of every digital control loop which has the same analog dynamics and compensator program that was tested, regardless of the output range or quantization size of the control loop. This holds true with the provision that the output variable has an output range at least equal to the number of quanta traveled by the output of the test loop in a given test.

If quantization is ignored, both the test loop and the digital control loop of Figure 1.3 can be modeled by the sampled-data system shown in Figure 2.1. Most of the results from sampled-data theory which will be used to synthesize transfer functions for the compensator have been presented in Section 2.1

and Chapter Four. All that remains is to explain the manner in which transfer function for the analog dynamics is obtained. A more detailed explanation of the following results may be found in reference 24.

The Laplace transform of the computation delay, zero-order hold network, and analog plant is given by

$$\left[e^{-(1-\Delta)Ts}\right]\left[\frac{1-e^{-Ts}}{s}\right]P(s)\ .$$

The first term arises from the computation delay $(1-\Delta)T$, the second term is the Laplace transform of the zero-order hold, and P(s) is the Laplace transform of the analog plant. AG(z) is the z- transform of the number sequence obtained by sampling, with the sample time T, the function of time that has the above Laplace transform. The result is:

$$AG(z) = z^{-1}(1-z^{-1})\ F_\Delta(z)\ , \tag{5.1}$$

where $F_\Delta(z)$ is the z- transform of the number sequence obtained by sampling the function of time which has the Laplace transform

$$\frac{e^{\Delta Ts}P(s)}{s}\ .$$

$F_\Delta(z)$ is called the advanced z- transform corresponding to P(s)/s and can be obtained from tables (e.g., ref. 24).

The parameter Δ will be approximately zero when the compensator is not operated with a high sampling rate. If Δ

is zero, AG(z) becomes:

$$AG(z) = z^{-2}(1 - z^{-1})\ F(z), \qquad (5.2)$$

where F(z) is the z- transform of the number sequence obtained by sampling the time function having the Laplace transform P(s)/s. F(z) is an ordinary z- transform and may also be obtained from tables (e. g., ref. 24). Since the ordinary z- transform corresponding to a given P(s)/s is considerably simpler than the advanced z- transform corresponding to P(s)/s, we shall employ the ordinary z- transform and use a signal from the compensator to control the sampling process so that Δ is exactly zero in those cases where the Δ given by equation (3. 6) is almost zero.

5. 2 First Order System with Integration

Wide bandwidth servo motors can often be accurately characterized by the following Laplace transfer function:

$$P(s) = \frac{K_A}{s(\tau s + 1)} \qquad (5.3)$$

This is the first analog plant that we shall attempt to compensate. K_A is assumed to be a combined gain constant which defines all of the gain through the analog dynamics. This system is used as an example to illustrate the behavior of the compensator with a variety of programs synthesized by methods presented in Chapter Four.

Let us first attempt to design a control loop with a ten cycle per second bandwidth. In Section 2. 2 it was decided that a sample time of four milliseconds should be used. A sample time of T = 3. 968 milliseconds was therefore selected from Table 3. 1. A motor time constant of 15 milliseconds is reasonable for operation in this frequency range. In order to simplify future arithmetic, τ was assumed to be exactly 4T:

$$\tau = 4T = 15.872 \text{ milliseconds.}$$

Examination of equation (3. 6) will show that a computation delay of close to one sample time must be accepted, since p = 2. Increasing the delay to exactly one sample time period therefore should not seriously degrade system performance, especially since the sample time is quite small. This was done so that the use of the simpler, ordinary z- transform could be illustrated.

The z- transform corresponding to P(s)/s is found from tables to be:

$$F(z) = K_A \left[\frac{Tz^{-1}}{1 - z^{-1}} - \frac{\tau (1 - e^{-T/\tau})z^{-1}}{(1 - e^{-T/\tau} z^{-1})} \right] \qquad (5.4)$$

AG(z) is found by substituting equation (5. 4) into equation (5. 2). After placing terms over a common denominator, we obtain:

$$AG(z) = K_A A_o \tau \left[\frac{z^{-2}(1 + A_1 z^{-1})}{(1 - z^{-1})(1 - e^{-T/\tau} z^{-1})} \right] , \qquad (5.5)$$

$$A_o = \frac{T}{\tau} - 1 + e^{-T/\tau} ,$$

$$A_1 = \frac{1}{A_o} \left[1 - e^{-T/\tau} (1 + T/\tau) \right] .$$

Since $(T/\tau) = 0.25$, we obtain:

$$A_o = 0.02880,$$

$$A_1 = 0.9201 , \qquad (5.6)$$

$$G(z) = \frac{z^{-2}(1 + 0.9201 \ z^{-1})}{(1 - z^{-1})(1 - 0.7788z^{-1})} .$$

A closed loop transfer function, K(z), must now be synthesized. Since the only zero of G(z) lies within the unit circle, the zeros of K(z) need not be restricted. Since j = 2, K(z) is of the following form:

$$K(z) = z^{-2} \sum_{i=0}^{J} \gamma_i \ z^{-i} . \qquad (5.7)$$

Let us constrain K(z) so that it will pass a ramp input with no steady state error. Then constraints (4.13) must be satisfied for q = 1:

$$\sum_{i=0}^{J} \gamma_i = 1$$

$$\sum_{i=0}^{J} i\gamma_i = -j = -2 \qquad (5.8)$$

The choice of J is somewhat arbitrary. Clearly the smallest value that can be used is J = 1, which yields $\gamma_1 = -2$, $\gamma_0 = 3$. This K(z) is called a "minimal prototype response function",[21,24] and yields the fastest response that can be obtained. Note from equation (4.28), however, that the step response overshoots 200 per cent. Also, the system is very noise sensitive with a variance reduction factor of 13. Past experience with sampled-data systems indicates that a variance reduction factor much larger than one should not be used.

The minimum response function is therefore not very desirable. In general, minimum response functions are rarely used in practice because of the large overshoots they exhibit, because they place no emphasis upon noise reduction, and because they usually require high loop gains. We shall therefore not consider them further in this report, especially since the sample time of the compensator was selected with the viewpoint that approximately four to eight samples would be allowed for the output to settle (see Section 2.2).

Let us try J = 4. This will yield a settling time of approximately one motor time constant and leave two degrees of freedom for noise reduction. The K(z) with the minimum variance reduction factor subject to constraints (5.8) is obtained from equation (4.16). Since $\alpha = j = 2$, we obtain

$$\gamma_i = 1 - \frac{2}{5} i; \quad i = 0, 1, 2, 3, 4. \tag{5.9}$$

Therefore,

$$K(z) = z^{-2} (1 + 0.6z^{-1} + 0.2 z^{-2} - 0.2 z^{-3} - 0.6z^{-4}). \tag{5.10}$$

The variance reduction factor is now

$$\sum_{i=0}^{J} \gamma_i^2 = 9/5 . \tag{5.11}$$

The maximum output with a unit step input, c_{SMAX}, is obtained from equation (4.28):

$$c_{SMAX} = \gamma_o + \gamma_1 + \gamma_2 = 1.8 . \tag{5.12}$$

Although the variance reduction factor and step response overshoot are much smaller than for the minimum response function, they are still too large for most applications. However, the program is realistic and was tested. The compensator transfer function is obtained from equation (2.3):

$$D(z) = \frac{K(z)}{AG(z)\,(1 - K(z))}$$

$$= \frac{(1-0.7788\,z^{-1})}{K_A \tau A_o\, z^{-2}\,(1 + 0.9201\,z^{-1})} \left[\frac{z^{-2} \sum_{i=0}^{4} \gamma_i\, z^{-i}}{\dfrac{(1 - z^{-2} \sum_{i=0}^{4} \gamma_i\, z^{-i})}{(1 - z^{-1})}} \right] \qquad (5.13)$$

We know that $1 - z^{-1}$ divides evenly into $1 - K(z)$ since $K(z)$ was designed to follow a step input with no steady state error. When this division and the indicated multiplications are performed, we obtain:

$$D(z) = \frac{1}{K_A \tau A_o} \left[\frac{1-0.179\,z^{-1} - 0.267\,z^{-2} - 0.356\,z^{-3} - 0.444\,z^{-4} + 0.467\,z^{-5}}{1+1.92\,z^{-1} + 0.920\,z^{-2} - 0.600\,z^{-3} - 1.35\,z^{-4} - 1.34\,z^{-5} - 0.552\,z^{-6}} \right] \qquad (5.14)$$

The digital gain is obtained from rule 3 in Section 3.8:

$$\frac{|b_1|}{K_D} = \frac{1.92}{K_D} \leq 1, \quad K_D \geq 1.92\,.$$

Since $K_D = 2(2^{K-1})$, we obtain $K_D = 2$, $K = 3$. The analog gain is given by rule 4:

$$\frac{|a_o|}{AK_D} = \frac{1}{A\,K_D} = 1\,.$$

Since $A = K_A \tau A_o$, we obtain

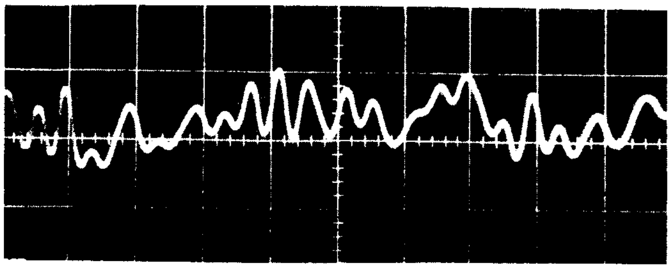
$$K_A = \frac{1}{\tau A_o K_D} = \frac{1}{(0.0159)(0.0288)(2)},$$

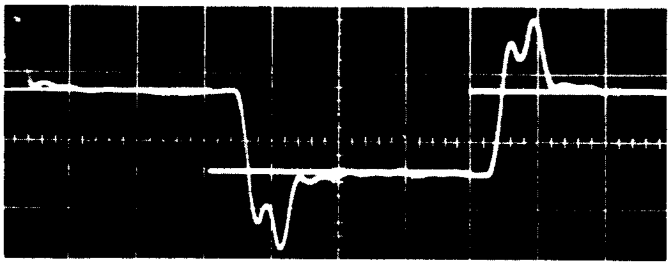
$$K_A = 1100.$$

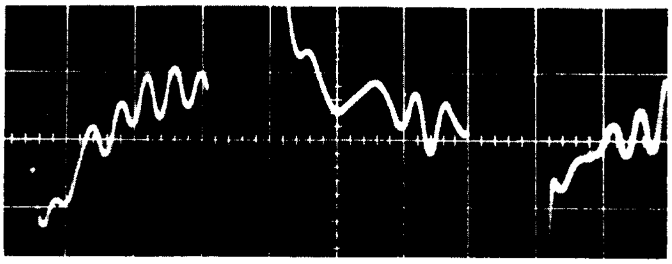
Finally, the constants K_i which are inserted into the diode matrix, are obtained from equations (3.13) and (3.14). They are listed below in decimal and binary form:

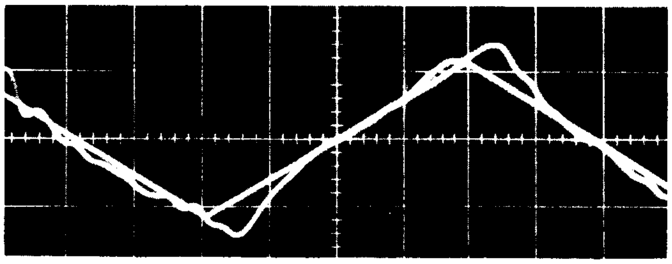
K_{15}	=	-0.960	=	-0.1111011
K_{14}	=	-0.460	=	-0.0111011
K_{13}	=	+0.300	=	+0.0100110
K_{12}	=	+0.676	=	+0.1010111
K_{11}	=	+0.668	=	+0.1010110
K_{10}	=	+0.276	=	+0.0100011
K_9	=	+1	=	+1
K_8	=	-0.179	=	-0.0010111
K_7	=	-0.267	=	-0.0100010
K_6	=	-0.356	=	-0.0101110
K_5	=	-0.444	=	-0.0111001
K_4	=	+0.467	=	+0.0111100

The experimental response curves are shown in Figure 5.3. Most of the figures illustrating experimental responses in this chapter will use the same format:


A. ERROR WITH, ZERO INPUT
2 Quanta/cm.
20 Msec./cm.


B. STEP RESPONSE
40 Quanta/cm.
20 Msec./cm.


C. STEP RESPONSE ERROR
2 Quanta/cm.
20 Msec./cm.


D. RAMP RESPONSE
10 Quanta/cm.
20 Msec./cm.

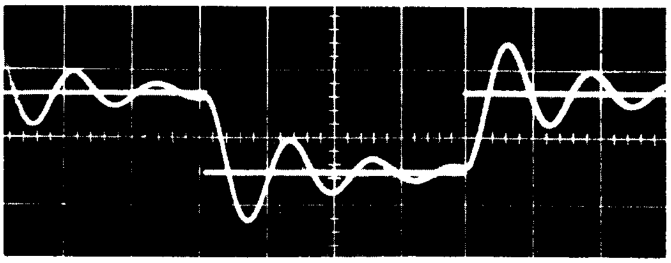

E. ANALOG RESPONSE
40 Quanta/cm.
20 Msec./cm.

FIGURE 5.3— EXPERIMENTAL RESPONSES FOR A FIRST ORDER SYSTEM WITH INTEGRATION, PROGRAM ONE

Photograph (a) shows the error with zero input.

Photograph (b) shows a step input and output.

Photograph (c) shows in great detail the steady state error present in the loop with the step input shown in (b). The large transient errors are off the scale in this photograph.

Photograph (d) shows a small amplitude ramp input and output.

Photograph (e), when present, shows the step input and output of a conventional analog, uncompensated control loop with the same open loop gain and analog plant as was used in the digitally compensated loop.

In all the photographs, zero amplitude is defined by the cross-marked horizontal grid line.

The ripple of slightly more than two quanta is surprisingly small considering that no emphasis was placed on the elimination of ripple other than minimizing the variance reduction factor. With zero input, this ripple varies about a DC error of approximately one quantum. An excellent example of how offset errors of this type can arise in quantized closed loop systems is given on page 364 of reference 21. Errors of this type were found to exist in most of the systems that were tested. In every case, however, the offset error remained constant with variations in the input amplitude. This error can therefore be subtracted from the input data at the time of preparation and should not be a matter for concern.

The step response exhibits the overshoot that was predicted, but the oscillation present in the overshoot represents a deviation from the theoretical response. The steady state error is zero if the one quantum offset is ignored.

The one quantum offset is barely discernible in the photograph of the ramp response.

The settling time is theoretically $6T = 23.8$ milliseconds. Both the ramp and step responses settle in exactly this time.

Let us now attempt to synthesize a program which will respond to a step input with zero steady state error and with no overshoot. The closed loop transfer function shall be constrained to have one zero equal to the zero of $G(z)$ in order to minimize ripple. The steady state ramp response will not be defined, but note from equation (4.14b) that the steady state ramp error will never go to infinity. In the steady state, the output will lag behind a ramp input by an amount defined by β, where β is obtained from equation (4.14b) once the constants $\{\gamma_i\}$ are fixed.

This problem is easily solved with the technique presented in Section 4.6. First, an output is defined which approaches the input smoothly, as shown in Figure 5.4. Since $j = 2$, the output must be zero at time T but will assume some nonzero value at time $2T$. This value defines the constants γ_o and

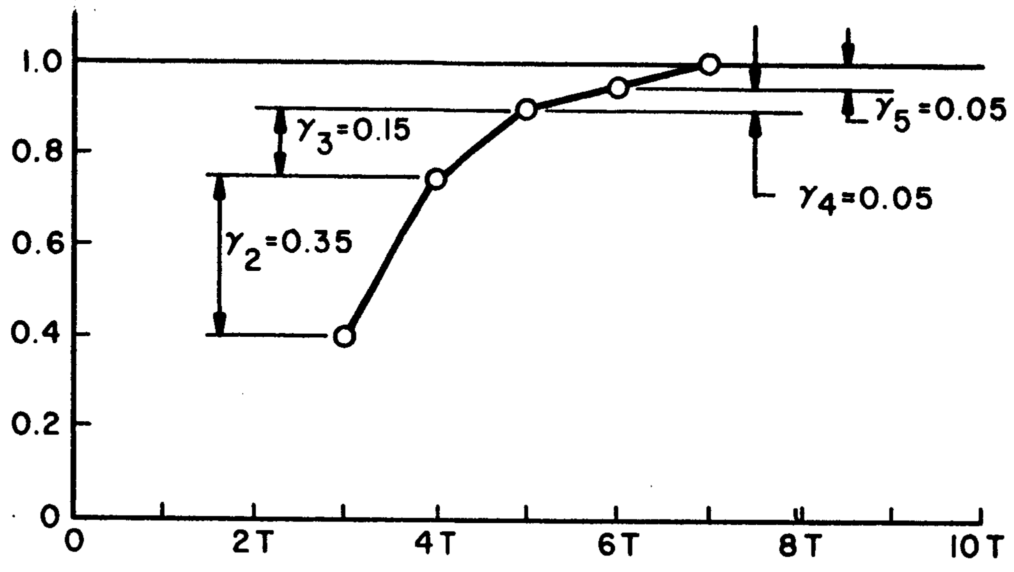


FIGURE 5.4– DESIRED FINAL APPROACH OF THE STEP RESPONSE

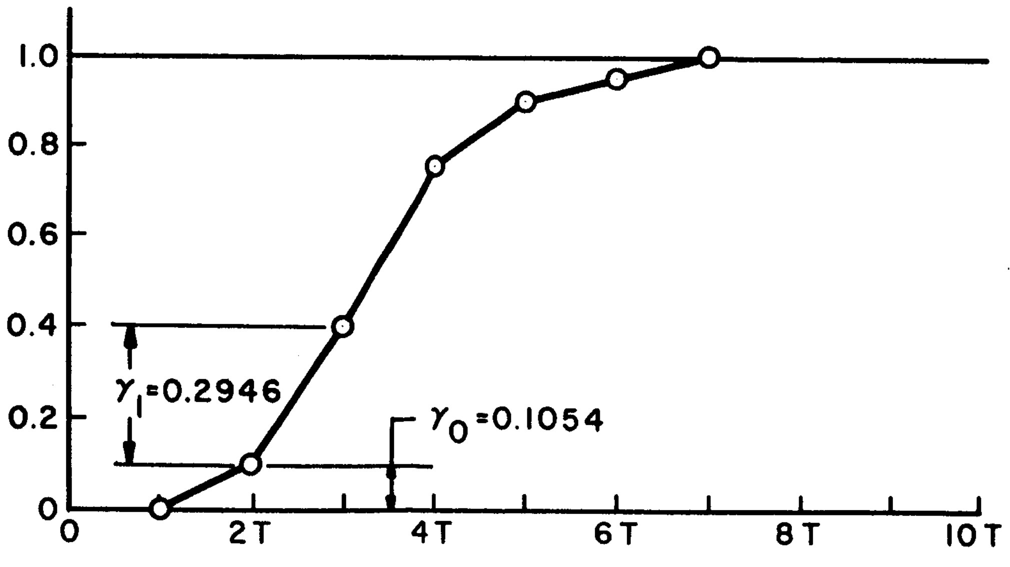


FIGURE 5.5– COMPLETED STEP RESPONSE FOR "RIPPLE FREE" OUTPUT

γ_1, and must be selected so that constraint (4.14a) is satisfied and so that K(z) has the same zero as G(z).

The constraint equation that the constants $\{\gamma_i\}$ must satisfy for K(z) to have the required zero is obtained by dividing K(z) by the numerator of G(z):

$$\frac{K(z)}{1 + A_1 z^{-1}} = \frac{z^{-j} \sum_{i=0}^{J} \gamma_i z^{-i}}{1 + A_1 z^{-1}} = z^{-j} \sum_{i=0}^{J-1} \beta_i z^{-i} + \frac{z^{-j}\left[\sum_{i=0}^{J} (-A_1)^{J-i} \gamma_i\right] z^{-J}}{1 + A_1 z^{-1}} \tag{5.15}$$

The constraint equation is obtained by forcing the remainder to be zero:

$$\sum_{i=0}^{J} (-A_1)^{J-i} \gamma_i = 0 \tag{5.16}$$

This equation and equation (4.14a) define γ_o and γ_1. When γ_o is substituted from equation (4.14a) into equation (5.16), we obtain

$$\gamma_1 = \frac{1 - \sum_{i=2}^{J} \gamma_i + \sum_{i=2}^{J} (-A_1)^{-i} \gamma_i}{1 + A_1^{-1}} \tag{5.17}$$

and γ_o is obtained from equation (4.14a):

$$\gamma_o = 1 - \gamma_1 - \sum_{i=2}^{J} \gamma_i \; .$$

Insertion of the constants which have been defined in Figure 5.4 into equations (5.17) and (4.14a) yield $\gamma_1 = 0.2946$ and $\gamma_o = 0.1054$. The completed step response therefore appears as in Figure 5.5.

In this case we were fortunate in obtaining a completely smooth response. If this had not occurred, the final portion of another acceptable output trajectory would have to be tried. After a few trials, one can usually develop an insight into the form that the step response must assume if K(z) is to satisfy the required constraints. By trying two extreme responses within the class of acceptable step responses, one can readily ascertain whether or not it is possible to obtain an acceptable step response while simultaneously satisfying all of the constraints which have been imposed upon K(z).

We now have K(z):

$$K(z) = z^{-2}(0.1054 + 0.2946\, z^{-1} + 0.35\, z^{-2} + 0.15\, z^{-3} + 0.05\, z^{-4} + 0.05\, z^{-5}), \tag{5.18}$$

or alternatively,

$$K(z) = z^{-2}\,(1 + 0.9201\, z^{-1}) \sum_{i=0}^{4} \beta_i\, z^{-i}. \tag{5.19}$$

The long division process (equation 5.15) yields the following relations for the constants $\{\beta_i\}$:

$$\beta_o = \gamma_o = 0.1054,$$
$$\beta_1 = \gamma_1 - A_1\gamma_o = 0.1976,$$
$$\beta_2 = \gamma_2 - A_1\beta_1 = 0.1682,$$
$$\beta_3 = \gamma_3 - A_1\beta_2 = -0.004719,$$
$$\beta_4 = \gamma_4 - A_1\beta_3 = 0.05434.$$

We can now find D(z):

$$D(z) = \frac{K(z)}{AG(z)\,(1 - K(z))},$$

$$D(z) = \frac{z^{-2}(1+0.9201\,z^{-1})\sum_{i=0}^{4}\beta_i\,z^{-i}}{K_A\tau A_o\left[\frac{z^{-2}(1+0.9201\,z^{-1})}{(1-z^{-1})(1-0.7788z^{-1})}\right]\left[1 - z^{-2}\sum_{i=0}^{5}\gamma_i\,z^{-i}\right]} \qquad (5.20)$$

Again, noting that $(1-z^{-1})$ must divide evenly into 1-K(z), we obtain:

$$D(z) = \frac{1}{K_A\tau A_o}\left[\frac{(0.105 - 0.116\,z^{-1} + 0.0142\,z^{-2} - 0.136\,z^{-3} + 0.0580\,z^{-4} - 0.0423\,z^{-5})}{(1 + z^{-1} + 0.895\,z^{-2} + 0.600\,z^{-3} + 0.250\,z^{-4} + 0.100\,z^{-5} + 0.0500\,z^{-6})}\right] \qquad (5.21)$$

It now follows (see previous example) that:

$$K_D = 1,$$

$$1 = 2\cdot 2^{3-K} = 2^{4-K},$$

$$K = 4.$$

Also, $$\frac{0.136}{K_A A_o \tau K_D} = 1,$$

$$K_A = \frac{0.136}{A_o\tau(1)} = 296.$$

The constants $\{K_i\}$ are now listed:

K_{15}	=	-1	=	-1
K_{14}	=	-0.895	=	-0.1110011
K_{13}	=	-0.600	=	-0.1001101
K_{12}	=	-0.250	=	-0.0100000
K_{11}	=	-0.100	=	-0.0001101
K_{10}	=	-0.050	=	-0.0000110
K_9	=	+0.779	=	+0.1100100
K_8	=	+0.853	=	+0.1101101
K_7	=	+0.105	=	+0.0001101
K_6	=	-1	=	-1
K_5	=	+0.428	=	+0.0110111
K_4	=	-0.312	=	-0.0100111

Before examining the experimental responses, let us first note some expected results. For step and ramp inputs, the output should reach its steady state value in 7T seconds or approximately 28 milliseconds. The amount that the ramp response will lag the input is calculated from equation (4.14b):

$$\beta = -j = -\sum_{i=0}^{5} i\,\gamma_i = -3.89 .$$

The ramp response should therefore lag the input by 3.89T seconds, or approximately 15.4 milliseconds.

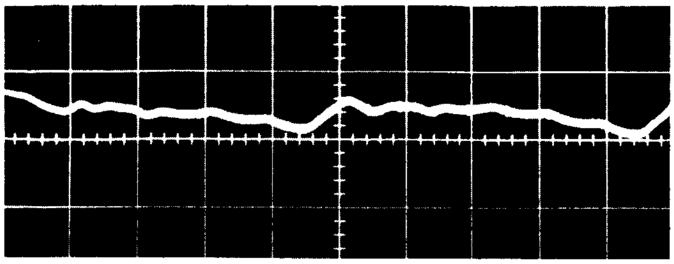
The variance reduction factor is $\sum_{i=0}^{5} \gamma_i^{\,2} = 0.248$. A variance reduction factor this low should result in excellent smoothing of the noise introduced by quantization. Since we have also designed K(z) to eliminate the ripple introduced by sampling, this program should yield about the smallest steady state ripple that can be obtained with this analog plant.

The experimental results are shown in Figure 5. 6. The steady state ripple is now well within two quanta. A DC offset of almost one quantum is again present with zero input.

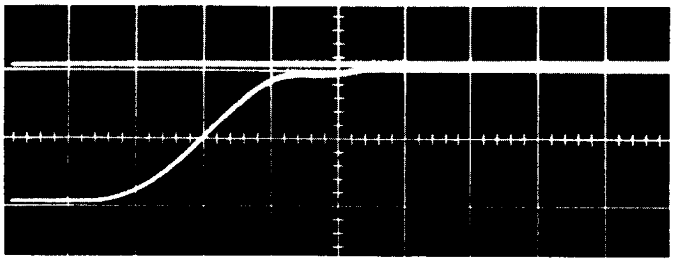
The step response is almost exactly as predicted by Figure 5. 5. (The step input was applied at the instant that the oscilloscope trace starts.) The detailed photograph of the step response error shows a steady state error of one quantum after removal of the one quantum offset. However, the error does stabilize at the one quantum offset level sixty milliseconds after application of the step input.

The ramp response lags the input by the predicted amount if the one quantum offset is removed. The response is also seen to settle in the predicted time.

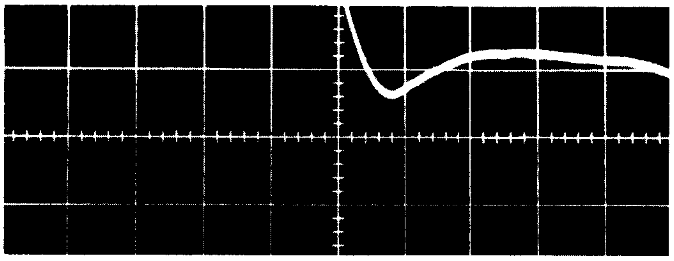
Let us now use the smallest sample time which can be programmed (T = 2. 048 milliseconds) in an attempt to further increase the system's bandwidth. The same analog plant shall be employed. Since the computation delay will now be significantly less than one sample time, the modified z- transform will be used to characterize the analog dynamics. (Use $\overline{S_{15-M}(B)}$ for the A/D input T_L as shown in Appendix B.)



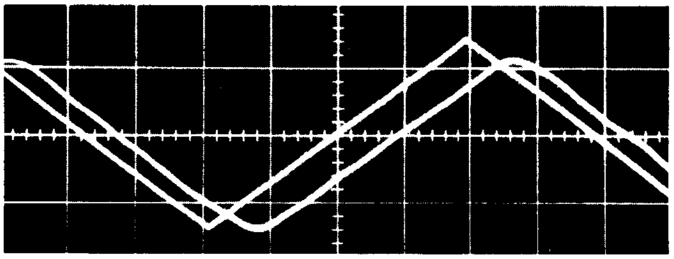
A. ERROR WITH ZERO INPUT
2 Quanta/cm.
20 Msec./cm.



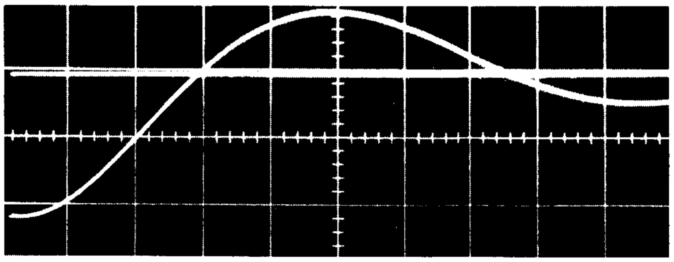
B. STEP RESPONSE
40 Quanta/cm.
5 Msec./cm.



C. STEP RESPONSE ERROR
2 Quanta/cm.
5 Msec./cm.



D. RAMP RESPONSE
10 Quanta/cm.
20 Msec./cm.



E. ANALOG RESPONSE
40 Quanta/cm.
5 Msec./cm.

FIGURE 5.6— EXPERIMENTAL RESPONSES FOR A FIRST ORDER SYSTEM WITH INTEGRATION, PROGRAM TWO

The advanced z- transform corresponding to P(s)/s is found from tables to be:

$$F_{\Delta}(z) = K_A \left[\frac{T z^{-1}}{(1 - z^{-1})^2} + \frac{a\Delta T - 1}{a(1 - z^{-1})} + \frac{e^{-a\Delta T}}{a(1 - e^{-aT} z^{-1})} \right], \qquad (5.22)$$

$$a = 1/\tau .$$

AG(z) is found by inserting this result into equation (5.1). After placing terms over a common denominator, we obtain:

$$AG(z) = K_A A_o z^{-1} \left[\frac{1 + A_1 z^{-1} + A_2 z^{-2}}{(1 - z^{-1})(1 - e^{-aT} z^{-1})} \right],$$

$$A_o = \frac{1}{a} (a\Delta T - 1 + e^{-a\Delta T}) ,$$

$$A_1 = \frac{1}{aA_o} \left[aT(1 - \Delta) + 1 + e^{-aT} (1 - a\Delta T) - 2 e^{-a\Delta T} \right], \qquad (5.23)$$

$$A_2 = \frac{1}{aA_o} \left[e^{-a\Delta T} + e^{-aT} (a\Delta T - 1 - aT) \right]$$

We must now find Δ. Examination of equation (3.6) will show that we must first fix M, the number of compensator outputs which must be remembered. This, in turn, requires us to define the form of K(z). Let us restrict K(z) to have the zeros of G(z) in order to reduce ripple:

$$K(z) = z^{-1} \sum_{i=0}^{J} \gamma_i z^{-i} = z^{-1} (1 + A_1 z^{-1} + A_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i}. \qquad (5.24)$$

When this equation is substituted into equation (2. 3) we obtain:

$$D(z) = \frac{1}{K_A A_o} \left[\frac{(1 - e^{-aT} z^{-1}) \sum_{i=0}^{J-2} \beta_i z^{-i}}{\frac{1 - z^{-1} \sum_{i=0}^{J} \gamma_i z^{-i}}{1 - z^{-1}}} \right] \tag{5.25}$$

If we assume that K(z) is designed to pass a step input with no steady state error, then $(1-z^{-1})$ divides evenly into $(1-z^{-1} \sum_{i=0}^{J} \gamma_i z^{-i})$. It therefore follows, by comparing with equation (2. 5) that $N = J-1$ and $M = J$. Substitution of these values into equation (3. 1) yields $2J \leq 15$, or $J_{MAX} = 7$. Let us take $J = 7$. Although this yields the longest settling time, it also yields the smallest variance reduction factor and step response overshoot. The settling time will still be small because the sample time is small.

Δ is now found by substituting $M = 7$ and $p = 1$ into equation (3. 6). We obtain $\Delta = 8/16 = 0.5$. The numerical values for the constants in equation (5. 23) may now be found:

$$AG(z) = K_A A_o z^{-1} \left[\frac{1 + A_1 z^{-1} + A_2 z^{-2}}{(1-z^{-1})(1-0.8789 z^{-1})} \right], \tag{5.26}$$

$$A_o = 3.233 \times 10^{-5}, \quad A_1 = 5.750, \quad A_2 = 0.9176.$$

Let us now further restrict K(z) to have the smallest possible variance reduction factor subject to the constraints that it pass a step and ramp input with no steady state error. The constants $\{\beta_i\}$ are then obtained from equation (4.23) with $J = 7$, $\alpha = j = 1$, $u_1 = A_1$ and $u_2 = A_2$. They are:

$$\beta_o = 0.1240\ , \quad \beta_1 = 0.04668\ , \quad \beta_2 = 0.03531\ ,$$
$$\beta_3 = 0.005000\ , \quad \beta_4 = -0.01093\ , \quad \beta_5 = -0.06960$$

The constants $\{\gamma_i\}$ are obtained from equation (5.24):

$$\gamma_o = 0.1239\ , \quad \gamma_1 = 0.7594\ , \quad \gamma_2 = 0.4174\ ,$$
$$\gamma_3 = 0.2509\ , \quad \gamma_4 = 0.05022\ , \quad \gamma_5 = -0.1279\ ,$$
$$\gamma_6 = -0.4102\ , \quad \gamma_7 = -0.06386.$$

D(z) is therefore given by:

$$D(z) = \frac{1}{K_A A_o}\left[\frac{\begin{array}{c}(0.124 - 0.0623\,z^{-1} - 0.00571\,z^{-2} - 0.0260\,z^{-3}\\ - 0.0153\,z^{-4} - 0.0600\,z^{-5} + 0.0612\,z^{-6})\end{array}}{\begin{array}{c}(1 + 0.876\,z^{-1} + 0.117\,z^{-2} - 0.301\,z^{-3} - 0.552\,z^{-4}\\ - 0.602\,z^{-5} - 0.474\,z^{-6} - 0.0637\,z^{-7})\end{array}}\right] \tag{5.27}$$

It follows that $K_D = 1(2^{3-K}) \geq 0.876$, $K = 3$, and $K_D = 1$. The analog gain is given by $(0.124/K_A A_o K_D) = 1$, or $K_A = 0.124/A_o = 3840$. The constants $\{K_i\}$ are:

$K_{15} =$	-0.876	$= -0.1110000$
$K_{14} =$	-0.117	$= -0.0001111$
$K_{13} =$	$+0.301$	$= +0.0100111$
$K_{12} =$	$+0.552$	$= +0.1000111$
$K_{11} =$	$+0.602$	$= +0.1001101$
$K_{10} =$	$+0.474$	$= +0.0111101$

K_9 = +0.0637	=	+0.0001000
K_8 = +1	=	+1
K_7 = -0.503	=	-0.1000000
K_6 = -0.0461	=	-0.0000110
K_5 = -0.210	=	-0.0011011
K_4 = -0.124	=	-0.0010000
K_3 = -0.484	=	-0.0111110
K_2 = +0.494	=	+0.0111111

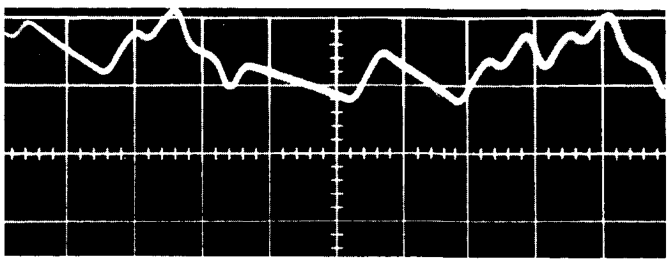
Theoretically, we have $c_{SMAX} = \sum_{i=0}^{4} \gamma_i = 1.602$, and $\sum_{i=0}^{J} \gamma_i^2 = 1.023$. The settling time should be 8T = 16.4 milliseconds.

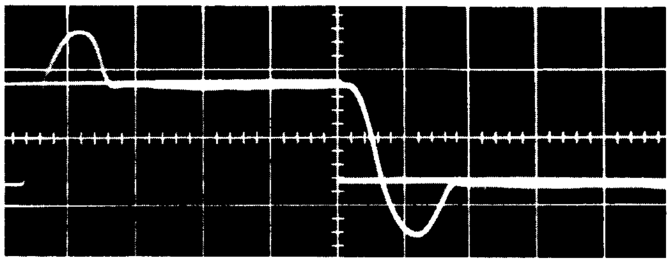
Figure 5.7 shows an offset error of three quanta in all of the responses. The settling time of the ramp and step response is exactly as predicted, as is the step response overshoot.

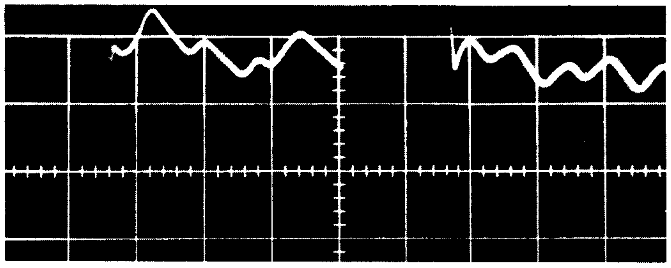
It is interesting to observe the improvement in step response and variance reduction factor that can be obtained by accepting a ramp response which lags the input by only one sample time period. A compensator program was synthesized in exactly the same manner as the previous program except that $\alpha = (-1 + j) = 0$ was used in equation (4.23) instead of $\alpha = 1$. The results are given below without further explanation.

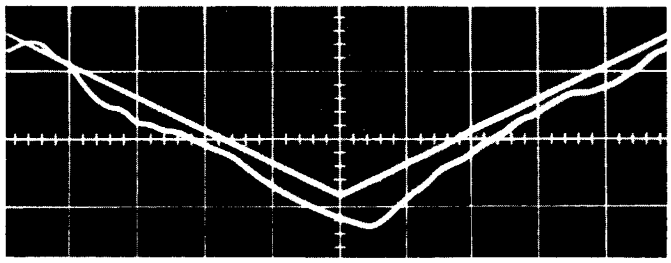
$\beta_o = 0.1024$, $\beta_1 = 0.04026$, $\beta_2 = 0.03194$

$\beta_3 = 0.008376$, $\beta_4 = -0.004517$, $\beta_5 = -0.04804$


A. ERROR WITH ZERO INPUT
2 Quanta/cm.
10 Msec./cm.


B. STEP RESPONSE
40 Quanta/cm.
10 Msec./cm.


C. STEP RESPONSE ERROR
2 Quanta/cm.
10 Msec./cm.


D. RAMP RESPONSE
10 Quanta/cm.
10 Msec./cm.

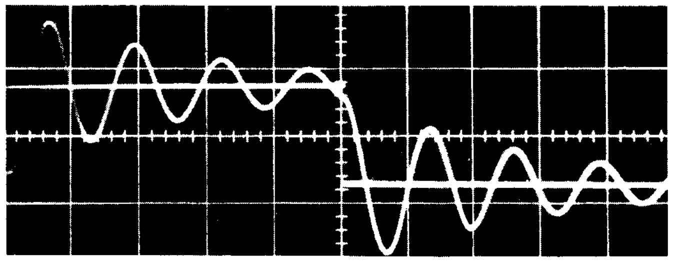

E. ANALOG RESPONSE
40 Quanta/cm.
10 Msec./cm.

FIGURE 5.7– EXPERIMENTAL RESPONSE FOR A FIRST ORDER SYSTEM WITH INTEGRATION, PROGRAM THREE

$\gamma_o = 0.1024$, $\gamma_1 = 0.6290$, $\gamma_2 = 0.3574$,

$\gamma_3 = 0.2289$, $\gamma_4 = 0.07295$, $\gamma_5 = -0.06633$,

$\gamma_6 = -0.2804$, $\gamma_7 = -0.04408$

$$D(z) = \frac{1}{K_A A_o} \left[\frac{(0.102 - 0.0497\,z^{-1} - 0.00345\,z^{-2} - 0.0197\,z^{-3} - 0.0119\,z^{-4} - 0.0441\,z^{-5} + 0.0422\,z^{-6})}{(1 + 0.898\,z^{-1} + 0.269\,z^{-2} - 0.0888\,z^{-3} - 0.318\,z^{-4} - 0.391\,z^{-5} - 0.324\,z^{-6} - 0.0439\,z^{-7})} \right] \quad (5.28)$$

$K_D = 1$, $K = 3$, $K_A = (0.102/A_o) = 3180$

K_{15}	=	-0.898	=	-0.1110011
K_{14}	=	-0.269	=	-0.0100010
K_{13}	=	+0.0888	=	+0.0001011
K_{12}	=	+0.318	=	+0.0101001
K_{11}	=	+0.391	=	+0.0110010
K_{10}	=	+0.324	=	+0.0101001
K_9	=	+0.0439	=	+0.0000110
K_8	=	+1	=	+1
K_7	=	-0.485	=	-0.0111110
K_6	=	-0.0337	=	-0.0000100
K_5	=	-0.192	=	-0.0011001
K_4	=	-0.116	=	-0.0001111
K_3	=	-0.431	=	-0.0110111
K_2	=	+0.412	=	+0.0110101

We note that now $c_{SMAX} = \sum_{i=0}^{4} \gamma_i = 1.391$, and $\sum_{i=0}^{J} \gamma_i^2 = 0.678$. The settling time is, of course, the same as before.

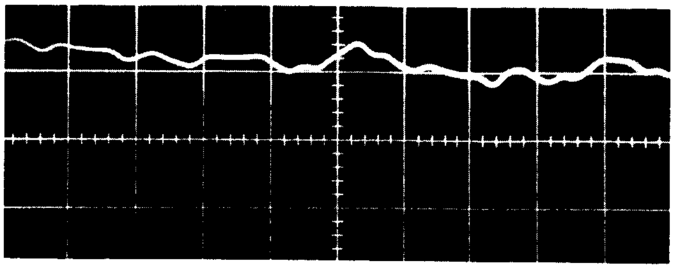
The experimental responses are shown in Figure 5.8. The offset error has increased, but this is of no consequence. The ripple is now smaller, as might be expected because of the smaller variance reduction factor. The step response overshoot has also decreased as predicted. Note that the settling time has remained unchanged and that the ramp response lags the input by one sample time period when the four quanta offset is removed.

5.3 Second Order Underdamped System

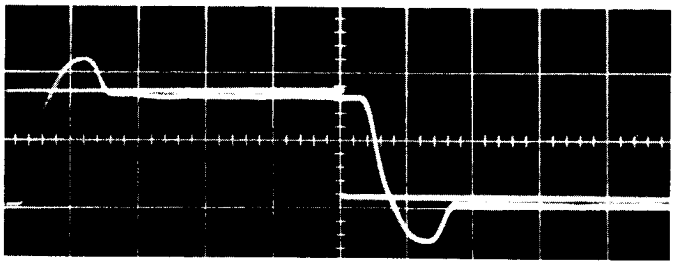
We shall now attempt to compensate a digital control loop which has the following analog plant:

$$P(s) = \frac{K_A \omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2}$$

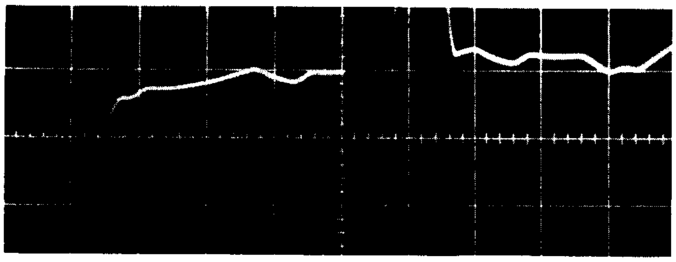
with $\zeta = 0.4$ and $\omega_n = 30.7$ (approximately five cycles per second). The sample time was selected from Table 3.1 to be 5.888 milliseconds, approximately one-fifth of the plant time constant $1/\omega_n$. Since $p = 3$, the computation delay will be approximately one sample time period. We therefore use the signal $\overline{S_1(W)}$ to control the A/D converter in order to have a computation delay exactly equal to the sample time. This enables the ordinary z- transform corresponding to P(s)/s to be used, resulting in



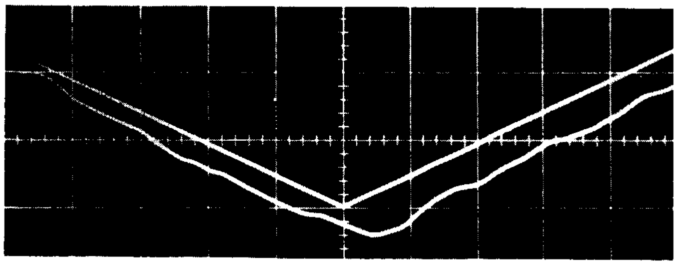
A. ERROR WITH ZERO INPUT
4 Quanta/cm.
10 Msec./cm.



B. STEP RESPONSE
40 Quanta/cm.
10 Msec./cm.



C. STEP RESPONSE ERROR
4 Quanta/cm.
10 Msec./cm.



D. RAMP RESPONSE
10 Quanta/cm.
10 Msec./cm.

FIGURE 5.8– EXPERIMENTAL RESPONSE FOR A FIRST ORDER SYSTEM WITH INTEGRATION, PROGRAM FOUR

the following pulse transfer function for the analog dynamics:

$$AG(z) = K_A A_o \left[\frac{z^{-2}(1 + A_1 z^{-1})}{1 + B_1 z^{-1} + B_2 z^{-2}} \right] ,$$

$$\begin{aligned} A_o &= e^{-aT} (e^{aT} - \cos bT - \tfrac{a}{b} \sin bT) , \\ A_1 &= e^{-aT} (e^{-aT} - \cos bT + \tfrac{a}{b} \sin bT)/A_o , \\ B_1 &= -2e^{-aT} \cos bT, \\ B_2 &= e^{-2aT} , \\ a &= \zeta \omega_n, \quad b = \omega_n \sqrt{1 - \zeta^2} . \end{aligned} \tag{5.29}$$

For our particular values we have:

$$\begin{aligned} A_o &= 0.01554 \\ A_1 &= 0.9529 \\ B_1 &= -1.835 \\ B_2 &= 0.8654. \end{aligned}$$

Since P(s) is underdamped and G(z) has a zero almost on the unit circle, K(z) had better be designed to have a zero equal to the one zero of G(z) in order to reduce ripple. K(z) is then of the form:

$$K(z) = z^{-2} \sum_{i=0}^{J} \gamma_i z^{-i} = z^{-2} (1 + A_1 z^{-1}) \sum_{i=0}^{J-1} \beta_i z^{-i} . \tag{5.30}$$

In anticipation of severe overshoots with an underdamped system, J was chosen as large as possible without exceeding the memory capacity of the compensator. The maximum J should yield the smoothest response. In order to find J_{MAX}, substitute K(z) into equation (2. 3). After simplification, we have

$$D(z) = \frac{(1 + B_1 z^{-1} + B_2 z^{-2}) \sum_{i=0}^{J-1} \beta_i z^{-i}}{K_A A_o (1 - z^{-2} \sum_{i=0}^{J} \gamma_i z^{-i})} . \qquad (5.31)$$

We therefore have N = J + 1 and M = J + 2. It follows from equation (3. 1) that $2J + 3 \leq 14$, or $J_{MAX} = 5$.

A minimum variance reduction factor program subject to constraints (5. 8) was then synthesized. The constants $\{\beta_i\}$ are obtained by solving equation (4. 24) for J = 5, $\alpha = 2$, and $u_1 = A_1$. The result is:

$$\beta_o = 0.7446 \quad , \quad \beta_1 = 0.001234 \quad , \quad \beta_2 = 0.1699 \; ,$$
$$\beta_3 = 0.0003436 \; , \quad \beta_4 = -0.4040.$$

The constants $\{\gamma_i\}$ are obtained from equation (5. 30):

$$\gamma_o = 0.7446 \quad , \quad \gamma_1 = 0.7107 \quad , \quad \gamma_2 = 0.1711 \; ,$$
$$\gamma_3 = 0.1622 \quad , \quad \gamma_4 = -0.4037 \quad , \quad \gamma_5 = -0.3850$$

We therefore have:

$$D(z) = \frac{1}{K_A A_o} \left[\frac{(0.745 - 1.365 z^{-1} + 0.812 z^{-2} - 0.310 z^{-3} - 0.258 z^{-4} + 0.742 z^{-5} - 0.350 z^{-6})}{(1 - 0.745 z^{-2} - 0.711 z^{-3} - 0.171 z^{-4} - 0.162 z^{-5} + 0.404 z^{-6} + 0.385 z^{-7})} \right] \qquad (5.32)$$

This program did not produce a satisfactory response in any sense. The loop would limit cycle unless it was started in a particular manner. When closed, the loop exhibited a ripple of over 20 quanta and would go into a limit cycle if disturbed slightly.

The reason for this occurrence is found by examining the final value of the unit step response of the compensator. This is obtained by letting z go to one in equation (5.32). The denominator, of course, goes to zero since it contains the factor $(1-z^{-1})^2$. The numerator goes to $0.016/K_A A_o$. This number is approximately zero in comparison with the numerator terms. Therefore, the numerator very nearly has the factor $(1-z^{-1})$, which corresponds to differentiation. The use of this program is therefore tantamount to placing two integrators in series with a differentiator. The differentiator tends to block the low frequency components of the error signal and they are never fully recovered by the integration. Due to inaccuracies in the differentiation and integration processes, the compensator appears as an approximate open circuit in the forward loop. Therefore, the loop always tends to drift away from its null position. In our particular case, the loop dynamics and compensator inaccuracies resulted in a continuous hunting about the null. In other cases, a program of this type could result in a system which would drift into saturation and remain there. This difficulty should not be blamed upon inaccuracies resulting from the particular instrumentation that was employed for the compensator. This type of program must be avoided with any form of compensation, whether it be analog or digital.

The limit cycle that was observed cannot be explained with the above simple argument because it is a nonlinear oscillation caused by saturation. With the inherent tendency of the loop to oscillate, however, the presence of the limit cycle is not surprising.

The difficulty with the above program is easily eliminated by dividing the numerator and denominator of D(z) by $(1-z^{-1})$. We have

$$\frac{\text{numerator}}{1-z^{-1}} = \frac{1}{K_A A_o} \left(0.745 - 0.620\, z^{-1} + 0.192\, z^{-2} - 0.118\, z^{-3} - 0.376\, z^{-4} + 0.336\, z^{-5} + \frac{0.016}{1-z^{-1}}\right) \tag{5.33}$$

and

$$\frac{\text{denominator}}{1-z^{-1}} = \left(1 + z^{-1} + 0.255\, z^{-2} - 0.455\, z^{-3} - 0.626\, z^{-4} - 0.789\, z^{-5} - 0.385\, z^{-6}\right) \tag{5.34}$$

If the remainder term is neglected, D(z) becomes:

$$D(z) = \frac{1}{K_A A_o}\left[\frac{(0.745 - 0.620\,z^{-1} + 0.192\,z^{-2} - 0.118\,z^{-3} - 0.376\,z^{-4} + 0.366\,z^{-5})}{(1 + z^{-1} + 0.255\,z^{-2} - 0.455\,z^{-3} - 0.626\,z^{-4} - 0.789\,z^{-5} - 0.385\,z^{-6})}\right] \quad (5.35)$$

The constants to be programmed are obtained as before.

$$K_D = 3(2^{3-K}) \geq 1, \quad K = 4, \quad K_D = 1.5, \quad \frac{0.745}{K_A A_o K_D} = 1,$$

$$K_A = \frac{0.745}{(0.01554)(1.5)} = 32.$$

K_{15}	=	-0.667	=	-0.1010110
K_{14}	=	-0.170	=	-0.0010110
K_{13}	=	+0.304	=	+0.0100111
K_{12}	=	+0.417	=	+0.0110101
K_{11}	=	+0.525	=	+0.1000011
K_{10}	=	+0.256	=	+0.0100001
K_9	=	+1	=	+1
K_8	=	-0.821	=	-0.1101001
K_7	=	+0.258	=	+0.0100001
K_6	=	-0.160	=	-0.0010100
K_5	=	-0.506	=	-0.1000001
K_4	=	+0.491	=	+0.0111111

The performance characteristics of interest for this program are $c_{SMAX} = \sum_{i=0}^{3} \gamma_i = 1.789$, and $\sum_{i=0}^{J} \gamma_i^2 = 1.433$. The settling time is 7T = 41.1 milliseconds.

The experimental results are shown in Figure 5.9. An offset error in all of the responses is again evident. The gross settling time of the step and ramp responses is 41 milliseconds as predicted, but the step response has a significant undershoot that should theoretically not be present. One hundred milliseconds are therefore required for the step response to truly settle. The step response overshoot is as predicted, and there is no steady state error in the step and ramp responses except for the offset error. Although it would probably be desirable to try other programs as was done in the previous section in order to reduce the step response overshoot, we shall not take the time to repeat these procedures here.

It is possible to derive an expression in terms of the constants present in the pulse transfer function of the analog dynamics which will yield an indication of when D(z) will have the difficulty that was discovered in this section. The difficulty will arise whenever the coefficients in the numerator polynomial defined by equation (2.3) add to approximately zero. If K(z) is of the form

$$K(z) = z^{-j} \left[1 + \sum_{i=0}^{Q} A_i z^{-i} \right] \left[\sum_{i=0}^{J-Q} \beta_i z^{-i} \right] \qquad (5.36)$$

and AG(z) is of the form

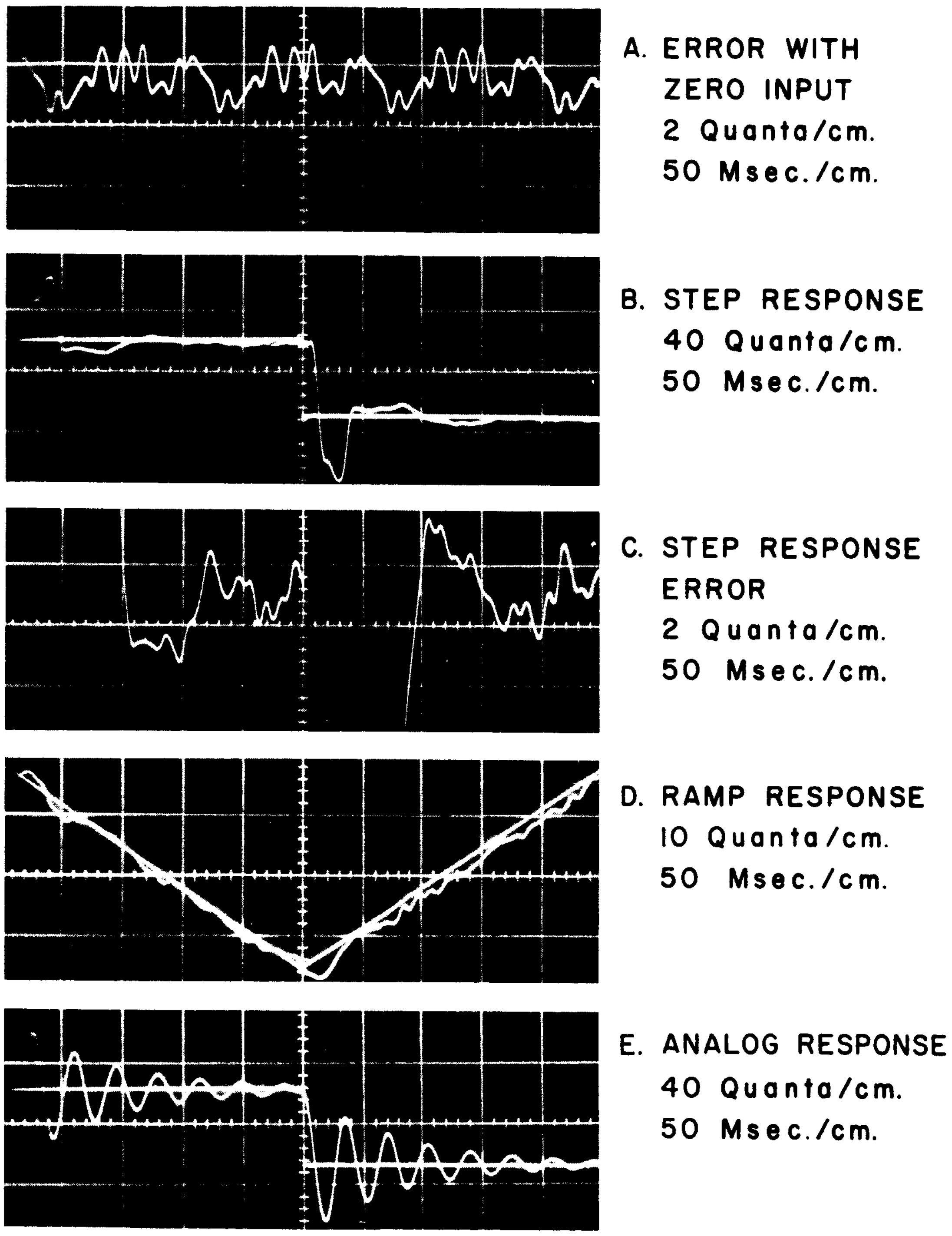


FIGURE 5.9— EXPERIMENTAL RESPONSES FOR A SECOND ORDER SYSTEM

$$AG(z) = A \left[\frac{z^{-j} \; (1 + \sum_{i=0}^{Q} A_i z^{-i})}{(1 - z^{-1})^m (1 + \sum_{i=1}^{L} B_i z^{-i})} \right] , \qquad (2.2)$$

where $(1 - z^{-1})^m$ is a factor of $1 - K(z)$, then it follows, by letting z go to one in equation (2.3), that the sum of the coefficients in the numerator polynomial of D(z) is

$$\sum_{i=0}^{N} a_i = \frac{1}{A} \left[1 + \sum_{i=1}^{L} B_i \right] \sum_{i=0}^{J-Q} \beta_i \quad . \qquad (5.37)$$

A slight extension of the development leading to equation (4.21a) yields:

$$\sum_{i=0}^{J-Q} \beta_i = \frac{1}{1 + \sum_{i=0}^{Q} A_i} \quad . \qquad (5.38)$$

When this result is inserted into equation (5.37), we obtain

$$\sum_{i=0}^{N} a_i = \frac{1}{A} \left[\frac{1 + \sum_{i=1}^{L} B_i}{1 + \sum_{i=0}^{Q} A_i} \right] . \qquad (5.39)$$

When K(z) is of the form

$$K(z) = z^{-j} \left[1 + \sum_{i=0}^{k} u_i z^{-i} \right] \sum_{i=0}^{J-k} \beta_i z^{-i} \quad , \qquad (5.40)$$

where $(1 + \sum_{i=0}^{k} u_i z^{-i})$ is a factor of the numerator of G(z), then

$$\sum_{i=0}^{N} a_i = \frac{1}{A}\left[\frac{1+\sum_{i=1}^{L} B_i}{1+\sum_{i=0}^{k} u_i}\right] . \tag{5.41}$$

Whenever $\sum_{i=0}^{N} a_i$ is small relative to the constants $\{A_i\}$ and $\{B_i\}$, $(1-z^{-1})$ will usually be an approximate factor of the numerator of $D(z)$.

5.4 Second Order Underdamped System with Integration

The compensation of a digital control loop with the analog plant

$$P(s) = \frac{K_A \omega_n^2}{s(s^2 + 2\zeta\omega n^s + \omega_n^2)} \tag{5.42}$$

shall now be illustrated. A natural frequency of $\omega_n = 12.56$ (approximately two cycles per second) was selected in order to illustrate the behavior of the compensator at its lower frequency limit. $\zeta = 0.4$ was again selected in order to demonstrate the more difficult underdamped case.

The largest possible sample time, T = 15.488 milliseconds was selected, which is approximately one-fifth of the time constant $1/\omega_n$. Since the computation delay will be nearly one sample time period, the A/D convetter was again controlled to provide a computation delay of exactly one sample time period. The ordinary z- transform could then be used to characterize the analog dynamics, resulting in

$$AG(z) = K_A A_o \left[\frac{z^{-2}\,(1 + A_1 z^{-1} + A_2 z^{-2})}{(1 - z^{-1})(1 + B_1 z^{-1} + B_2 z^{-2})} \right] ,$$

$$\begin{aligned}
A_o &= T - \frac{2\zeta}{\omega_n}\,(1 + B_1 + \sigma) , \\
A_1 &= \frac{1}{A_o}\left(T B_1 + \frac{2\zeta}{\omega_n}(B_1 - B_2 + 1 + 2\sigma)\right) , \\
A_2 &= \frac{1}{A_o}\left(T B_2 + \frac{2\zeta}{\omega_n}(B_2 - \sigma)\right) , \\
B_1 &= -2 e^{-\zeta\omega_n T} \cos(\omega_n T\sqrt{1 - \zeta^2}) , \\
B_2 &= e^{-2\zeta\omega_n T} \\
\sigma &= \frac{(-\zeta + 1/2\zeta)\, e^{-\zeta\omega_n T} \sin(\omega_n T\sqrt{1 - \zeta^2})}{\sqrt{1 - \zeta^2}} + \\
&\quad + e^{-\zeta\omega_n T} \cos(\omega_n T\sqrt{1 - \zeta^2}) .
\end{aligned} \tag{5.43}$$

For our particular values, we have:

$$\begin{aligned}
A_o &= 9.382 \times 10^{-5} \\
A_1 &= 3.842 \\
A_2 &= 0.9251 \\
B_1 &= -1.821 \\
B_2 &= 0.8559.
\end{aligned}$$

In order to obtain as little ripple as possible, let us use a K(z) of the form

$$K(z) = z^{-2} \sum_{i=0}^{J} \gamma_i z^{-i} = z^{-2}(1 + A_1 z^{-1} + A_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i} , \tag{5.44}$$

and use the largest possible J. As in the previous section, J_{MAX} is found by substituting K(z) into equation (2. 3):

$$D(z) = \frac{(1 + B_1 z^{-1} + B_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i}}{K_A A_o \left[\dfrac{1 - z^{-2} \sum_{i=0}^{J} \gamma_i z^{-i}}{1 - z^{-1}} \right]} . \tag{5.45}$$

Now note from equation (5. 39) that $\sum_{i=0}^{N} \alpha_i = \frac{1}{A} \frac{0.0349}{5.767} = \frac{1}{A}(.0065)$. Therefore, the numerator of the right hand side of equation (5. 45) will probably have the approximate factor $(1-z^{-1})$. In order to avoid this situation, the numerator and denominator in equation (5. 45) must be divided by $(1-z^{-1})$. This implies that $(1-z^{-2} \sum_{i=0}^{J} \alpha_i z^{-i})$ must have the factor $(1-z^{-1})^2$, and we must therefore constrain K(z) to satisfy equations (5. 8). After performing the required cancellations, it follows that N = J-1 and M = J. Therefore $2J-1 \leq 14$, or $J_{MAX} = 7$.

A minimum variance reduction factor program subject to the above constraints was then synthesized by substituting $J = 7$, $u_1 = A_1$, $u_2 = A_2$, and $\alpha = 2$ into equation (4. 23). The results follow:

$\beta_o = 0.2124$, $\beta_1 = -0.03845$, $\beta_2 = 0.06178$,
$\beta_3 = -0.008006$, $\beta_4 = -0.0002126$, $\beta_5 = -0.1310$

$\gamma_o = 0.2124$, $\gamma_1 = 0.8544$, $\gamma_2 = 0.4060$,
$\gamma_3 = 0.2649$, $\gamma_4 = 0.02619$, $\gamma_5 = -0.1392$,
$\gamma_6 = -0.5035$, $\gamma_7 = -0.1212$

$$(1 + B_1 z^{-1} + B_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i} = (0.2124 - 0.3483 z^{-1} + 0.1736 z^{-2} - 0.08759 z^{-3} + 0.06724 z^{-4} - 0.1375 z^{-5} + 0.2384 z^{-6} - 0.1121 z^{-7})$$

$$\frac{(1 + B_1 z^{-1} + B_2 z^{-2}) \sum_{i=0}^{J-2} \beta_i z^{-i}}{1 - z^{-1}} = (0.212 - 0.136 z^{-1} + 0.0376 z^{-2} - 0.0500 z^{-3} + 0.0173 z^{-4} - 0.120 z^{-5} + 0.118 z^{-6}) + \frac{0.006 z^{-7}}{1 - z^{-1}}$$

$$\frac{1 - z^{-2} \sum_{i=0}^{J} \gamma_i z^{-i}}{(1 - z^{-1})^2} = (1 + 2 z^{-1} + 2.79 z^{-2} + 2.72 z^{-3} + 2.25 z^{-4} + 1.51 z^{-5} + 0.746 z^{-6} + 0.112 z^{-7})$$

$$D(z) = \frac{1}{K_A A_o} \left[\frac{(0.212 - 0.136 z^{-1} + 0.0376 z^{-2} - 0.0500 z^{-3} + 0.0173 z^{-4} - 0.120 z^{-5} + 0.118 z^{-6})}{(1 + 2z^{-1} + 2.79 z^{-2} + 2.72 z^{-3} + 2.25 z^{-4} + 1.51 z^{-5} + 0.746 z^{-6} + 0.122 z^{-7})} \right]$$

$$K_D = 8(2^{3-K}) \geq 2.79, \qquad K = 4, \qquad K_D = 4,$$

$$\frac{0.212}{A_o K_A K_D} = 1, \qquad K_A = \frac{0.212}{(9.38 \times 10^{-5})(4)} = 567.$$

K_{15}	$= -0.5$	$= -0.1000000$
K_{14}	$= -0.698$	$= -0.1011001$
K_{13}	$= -0.680$	$= -0.1010111$
K_{12}	$= -0.563$	$= -0.1001000$
K_{11}	$= -0.378$	$= -0.0110000$
K_{10}	$= -0.187$	$= -0.0011000$
K_9	$= -0.0304$	$= -0.0000100$
K_8	$= +1$	$= +1$
K_7	$= -0.639$	$= -0.1010010$
K_6	$= +0.177$	$= +0.0010111$
K_5	$= -0.235$	$= -0.0011110$
K_4	$= +0.0814$	$= +0.0001010$
K_3	$= -0.566$	$= -0.1001000$
K_2	$= +0.557$	$= +0.1000111$

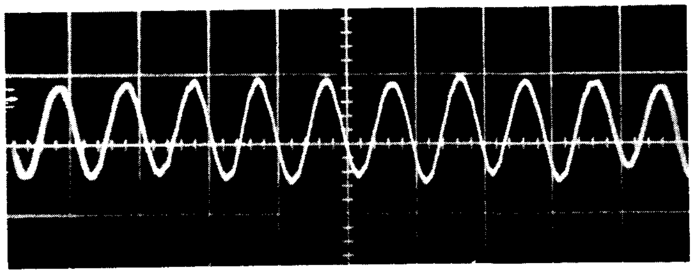
$$c_{SMAX} = \sum_{i=0}^{4} \gamma_i = 1.764$$

$$\sum_{i=0}^{J} \gamma_i^2 = 1.301$$

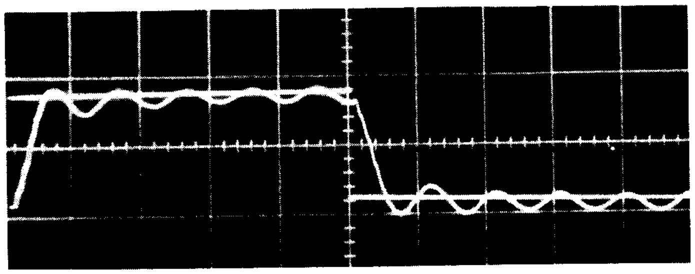
Settling time = 9T = 0.139 seconds

The experimental responses are shown in Figure 5.10. The first serious problem associated with saturation of the compensator was encountered. The step input shown in Figure 5.10a is the largest input that can be accepted before the output of the compensator saturates.

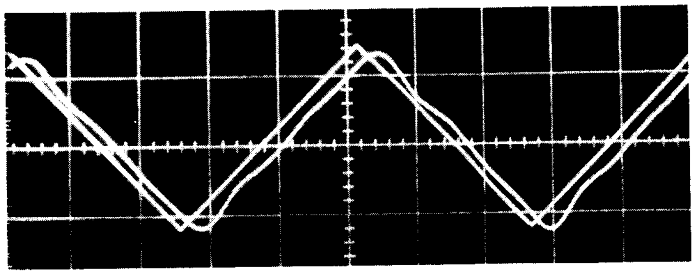
The first serious deviations of the experimental responses from the theoretical responses predicted by sampled-data theory have also been encountered. The step response does not overshoot and requires almost twice the time to settle as was predicted. The existence of this deviation should not be surprising. Since the input step is only sixteen quanta in amplitude, quantization errors can be expected to play a large part in shaping the response. In order to fully predict a response when quantization errors are significant, one would have to perform a point by point plot of the response, taking into account the quantization process associated with multiplication within the compensator as well as the quantization of the error. Such an analysis, however, would apply only to a particular compensation problem and would very likely yield no information as to how to alter a compensator program in order to improve a particular response. Studies have been made to predict the effects of quantization in closed loop systems, but with very limited success.[4,21] Only an upper bound on the bias error, which we have found to be relatively unimportant,



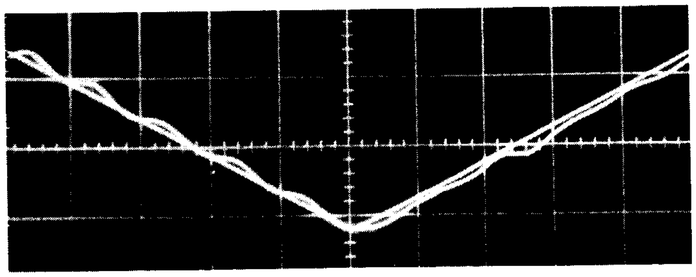
A. ERROR WITH ZERO INPUT
2 Quanta/cm.
0.5 Sec./cm.



B. STEP RESPONSE
10 Quanta/cm.
0.5 Sec./cm.



C. RAMP RESPONSE
10 Quanta/cm.
0.5 Sec./cm.



D. RAMP RESPONSE
10 Quanta/cm.
0.5 Sec./cm.

FIGURE 5.10— EXPERIMENTAL RESPONSES FOR A SECOND ORDER SYSTEM WITH INTEGRATION

can be obtained unless one wishes to perform a detailed point by point analysis.

The ramp response also shows a significant departure from the predicted response. Figure 5.10 shows a lag of 0.1 seconds in the ramp response. This lag is significant since it is almost equal to the theoretical settling time and the program was synthesized to yield a zero steady state ramp response error. Figure 5.10d was included to show that things are not always as bad as they seem. With an input rate of half that shown in Figure 5.11c, the output follows the input with negligible error.

No closed loop analog response is shown because the conventional analog loop was unstable at the gain that was employed.

5.5 Conclusions

The compensator was tested with three different forms of analog plants which are commonly encountered in practice. The time constants of the plants were selected so that the compensator was tested over its entire recommended frequency range of two to twenty cycles per second. Very satisfactory performance was obtained in all cases with compensator programs synthesized with sampled-data techniques. The correlation between actual responses and those predicted by sampled-data theory, where it is assumed that the variables are unquantized, was truly remarkable. Only with the last program, which was synthesized to compensate an underdamped second order system with integration, were any significant departures from the sampled-data theory responses noted. This type of analog plant, however, is close to being as difficult a system to compensate as will ever be encountered in practice. Although higher order systems are common, they will usually have "lead" terms in the numerator of the transfer function to help reduce the time lag through the analog dynamics. We can safely conclude that the objective of designing a compensator which can be applied in virtually every high performance digital servomechanism at a cost comparable to the components presently employed in these control loops was accomplished.

5.6 Recommendations for Improvement of the Compensator

In view of the excellent results that have been obtained, there is little to recommend in the way of improvement. There appears to be no need for increased computational accuracy and very little hardware can be eliminated by reducing accuracy. The number of samples of the variables which are remembered are adequate but not excessive. Fewer samples, and hence less memory and longer sample time, cannot be used because then minimal response functions, which are inadequate for most applications, would have to be employed. Of course, the possibility of using small memory and fast sample time so that the compensator operates as a continuous element always exists, but this would require an entirely new investigation.

The only suggestion for improvement is to decrease the input word size and increase the output word size. With the present design, the output of the compensator almost always saturates before the full input range of the compensator is utilized. The decrease in input word size can therefore be made without degrading the compensator's performance, and its output range can be increased by utilizing the memory vacated by the input data.

LIST OF REFERENCES

1. Analog-Digital Conversion Handbook, Digital Equipment Corporation, Maynard, Mass., 1964.

2. Beckett, J. T. and Mergler, H. W., "Analysis of an Incremental Digital Positioning Servosystem with Digital Rate Feedback", ASME Paper No. 64-WA/AUT-3, 1964.

3. Bennett, W. R., "Spectra of Quantized Signals", Bell System Technical Journal, Vol. 27, July, 1948, pp. 446-472.

4. Bertram, J. E., "The Effect of Quantization in Sampled-Feedback Systems", AIEE Transactions, Part II, Vol. 77, 1958, pp. 177-181.

5. Braun, E. L., "A Comparison of Integral and Incremental Digital Computers for Process Control Applications", Control Engineering, Jan. 1960, pp. 113-118.

6. Conn, R. B., "Digital Computers for Linear, Real-Time Control Systems", Proceedings of the Eastern Joint Computer Conference, 1953, pp. 33-37.

7. Curl, F. G., "A Comparison of Computers", Computers in Control Conference, AIEE Special Publication S-132, Sept. 1961, pp. 87-96.

8. Dickinson, M. M., "A Comparison of Digital Differential Analyzer and General Purpose Equipment in Guidance Systems", Computers in Control Conference, AIEE Special Publication S-132, Sept. 1961, pp. 208-210.

9. Gawlowicz, D. J., Taft, C. K. and Wijnschenk, M. J., "A Simple Digital Lead Compensator for Pulse-Data Servomechanisms", (To be published in Control Engineering), 1965.

10. Githens, J. A., "Digital Differential Analyzers: a Tutorial Paper", TRADIC Computer Research Program, Report for the 1st, 2nd, 3rd and 4th Quarters, Contract No. AF33 (600)-21536, Section 9, April 1, 1955.

11. Haggerty, P. E., et al., "Integrated Circuits", IEEE Spectrum, June 1964, pp. 62-82.

12. Ho, Y. C. and Johnson, E. C., "Design of a Numerical Milling Machine System", Proceedings of the Eastern Joint Computer Conference, 1957, pp. 11-24.

13. Hollander, G. L., "Transac C-1100: Transistorized Computers for Airborne and Mobile Systems", IRE Transactions on Aeronautical and Navigational Electronics, Sept. 1958, pp. 159-169.

14. Kearns, R. W., "A Digital Compensator for Automatic Control Systems", Ph. D. Thesis, Case Institute of Technology, 1964.

15. Linebarger, R. N., "Precision-Sample Rate Tradeoffs in Quantized Sampled Data Systems", Ph. D. Thesis, Case Institute of Technology, 1964.

16. Maley, G. A. and Earle, J., The Logic Design of Transistor Digital Computers, Englewood Cliffs, N. J.: Prentice-Hall, Inc., 1963.

17. Mergler, H. W., Space-Borne Digital Computers, Aerospace Systems Div., General Precision, Inc., Little Falls, N. J., Contract 3ST - 17175, 1962.

18. Mergler, H. W., et al., Digital Control Systems Engineering, Cleveland, Ohio: Case Institute of Technology, 1962.

19. Mergler, H. W. and Hubbard, K. H., "A Single-Loop, Two-Mode, Digital Process Controller", IEEE International Convention Record, 1965.

20. Meyer, M. A., "Digital Techniques in Analog Systems", IRE Transactions on Electronic Computers, vol. EC-3, June 1954, pp. 23-29.

21. Monroe, A. J., Digital Processes for Sampled-Data Systems, New York: John Wiley and Sons, Inc., 1962.

22. Moore, G. T., "The Numericord Machine-Tool Director", Proceedings of the Eastern Joint Computer Conference, 1957, pp. 6-10.

23. Peatman, J. B., "A Digital Controller Employing Truncated Logarithmic Quantization", Ph. D. Thesis, Case Institute of Technology, 1965.

24. Raggazini, J. R. and Franklin, G. F., Sampled Data Control Systems, New York: McGraw-Hill Book Co., 1958.

25. Roehr, W. D. and Thorpe, D., (Ed.) Motorola Switching Transistor Handbook, Motorola, Inc., 1963.

26. Sendzuk, G. T., "Results of Simulation Comparison of Control Computers", AIEE Transactions, pt. I, Vol. 80, Nov. 1961, pp. 547-550.

27. Sendzuk, G. T., "A Variable Increment Computer", Computers in Control Conference, AIEE Special Publication S-132, Sept. 1961, pp. 112-120.

28. Shackell, S. M. and Tryon, J. G., "The Relative Merits of Incremental and Conventional Digital Computers in Air-Borne Real-Time Control", AIEE Transactions, pt. I, Vol. 79, Sept. 1960, pp. 393-400.

29. Susskind, A. K., "Characteristics and Appraisal of Analog-Digital Conversion Techniques", Proceedings of the Computers in Control Systems Conference, AIEE Publication, 1957, pp. 26-29.

30. Taft, C. K. and Gaither, P. H., "Digital Lead Compensation of Pulse-Data Control Systems", ASME Paper No. 63-WA-183, Nov. 1963.

31. Van Pelt, R. W., "A Digital Controller Using Multirate Sampling for Gain Control", Ph. D. Thesis, Case Institute of Technology, 1965.

APPENDIX A
DIGITAL CIRCUITS AND CORRESPONDING LOGIC DIAGRAMS

Discrete component transistor circuits were used throughout the design. Although some 200KC flip-flops were used, all gating and most registers are composed of higher speed circuits. The 200 KC flip-flops will be explicitly labeled as such to distinguish them from the high speed flip-flops; the same logic symbol is used for the two types of flip-flops since they perform identical logic functions.

The 200 KC logic which was presently in use at the Digital Systems Laboratory at Case Institute of Technology, and the serial memory operated with zero and -6 volt output levels. Since commercial high speed circuits with these output levels were quite expensive, a line of high speed circuits was designed and built for use in the compensator. These circuits were tested at frequencies up to five megacycles. Judging from the probagation delays and capacitor recovery times, however, they should operate reliably at frequencies up to ten megacycles. The important characteristics of the circuits are tabulated in Table A. 1. The rise and fall of times of the circuits are less than 25 nanoseconds; no effort was made to obtain an accurate measurement because these times are usually not important in logic design.

LOAD (GATE LEGS)	PROBAGATION DELAY (NANO SECONDS)		MAXIMUM LOAD (GATE LEGS)
	TURN ON	TURN OFF	
1	9	9	7
5	9	30	

A. NOR GATE

LOAD (GATE LEGS)	PROBAGATION DELAY (NANO SECONDS)	MAXIMUM LOAD (GATE LEGS)
1	25	5
5	40	

B. FLIP–FLOP

TABLE A.1– 5 MC LOGIC CHARACTERISTICS

The 200 KC flip-flops have an unloaded switching time of 0.2 microseconds and will drive up to seven gate legs.

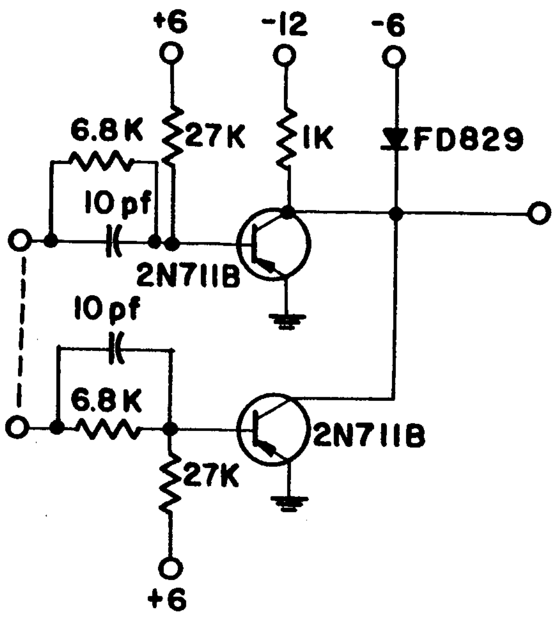
The circuits are shown in Figures A. 1 through A. 4. The gates and flip-flops are of a standard configuration and a description of their electronic behavior may be found in numerous sources.[25] Therefore, we shall only discuss their behavior as logical elements.

A logical "zero" is represented by zero volts (saturated transistor), and a logical "one is represented by a voltage less than -6 volts (cut off transistor).

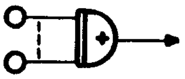
The operation of the NOR gate should be clear. The gated pulse generator will emit a "zero" to "one" pulse when the capacitor input switches from "one" to "zero" provided the other level is "zero".

The five megacycle and 200 KC flip-flops operate in exactly the same manner. Terminals E and C must always be grounded. They are shown explicitly since they may be used manually to set or reset the flip-flop. The flip-flop is set by opening pin C and reset by opening pin E. The flip-flop may also be manually set by grounding pin F and reset by grounding M.

Terminals S and R are DC set (S) and reset (R) inputs. Application of a "one" to one of these inputs will send the flip-flop to the appropriate state and hold it there as long as the "one" level is present, regardless of any transitions on inputs J and K.

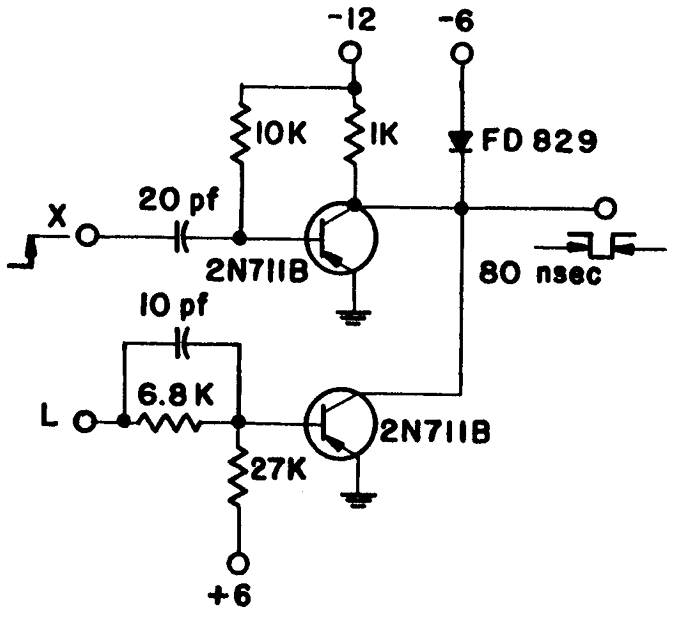


A. CIRCUIT DIAGRAM

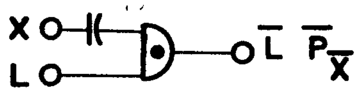


B. LOGIC DIAGRAM

FIGURE A.1– NOR GATE CIRCUIT AND LOGIC DIAGRAM (5 MC)

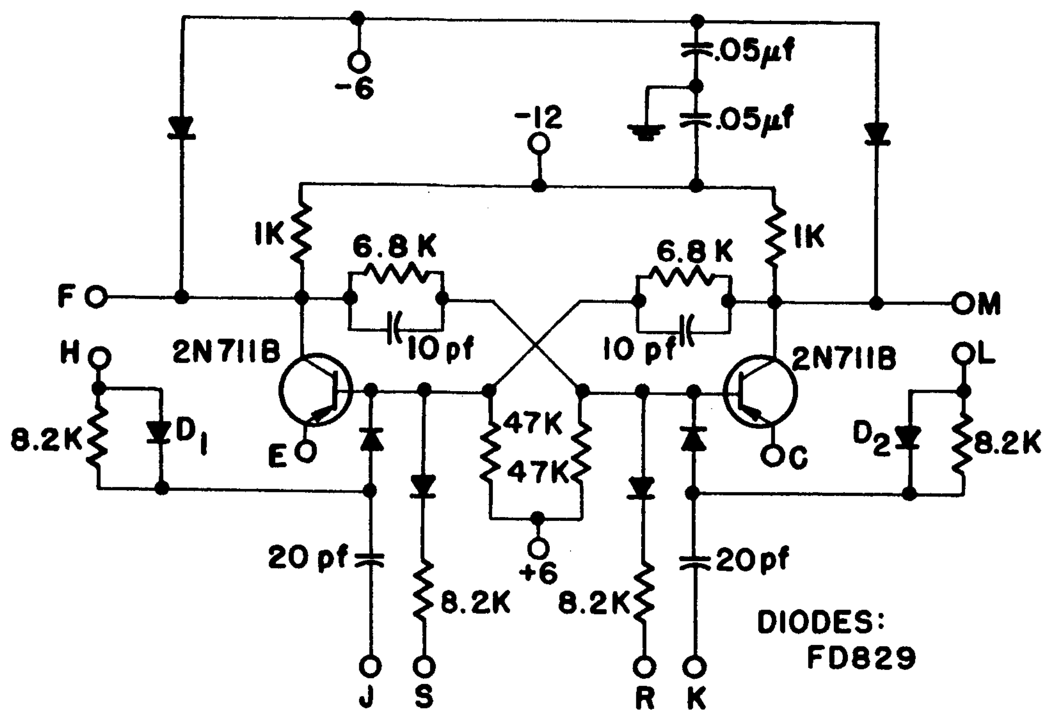


A. CIRCUIT DIAGRAM

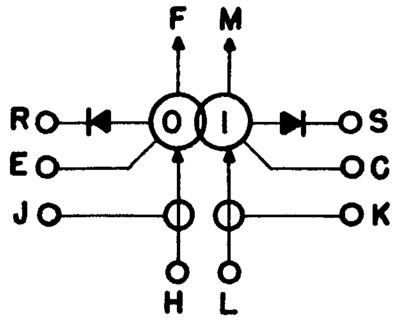


B. LOGIC DIAGRAM

FIGURE A.2— GATED PULSE GENERATOR CIRCUIT AND LOGIC DIAGRAM (5 MC)

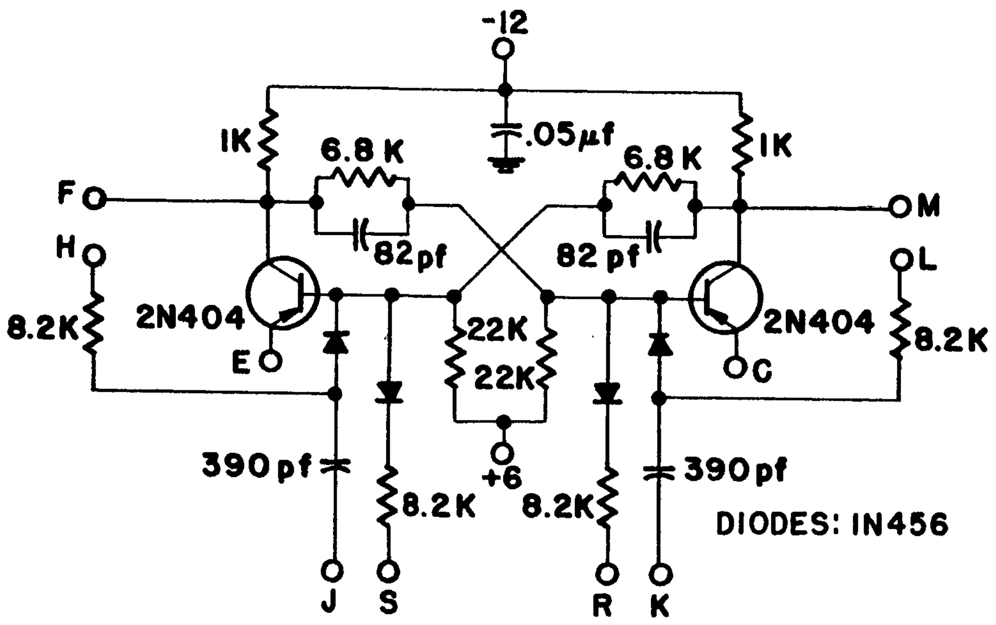


A. CIRCUIT DIAGRAM

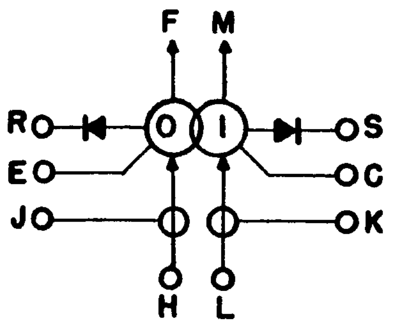


B. LOGIC DIAGRAM

FIGURE A.3- 5 MC FLIP-FLOP CIRCUIT AND LOGIC DIAGRAM



A. CIRCUIT DIAGRAM



B. LOGIC DIAGRAM

FIGURE A.4– 200 KC FLIP-FLOP CIRCUIT AND LOGIC DIAGRAM

Terminals H and L must be attached to control levels. These levels define whether or not "one" to "zero" transitions on pins J and K will switch the flip-flop. (A "zero" to "one" transition will never switch the flip-flop.) A transition on input J will reset the flip-flop if level H is "zero". Likewise, a transition on input J will reset the flip-flop if level H is "zero". Likewise, a transition on input K will set the flip-flop if L is "zero".

The flip-flop may be connected to operate as an AC set-reset flip-flop or as a trigger (change) flip-flop. Abbreviated logic symbols were used for these two configurations because of their widespread application. These logic symbols and the interconnections implied by them are defined in Figures A. 5 and A. 6.

Usually, all inputs to a flip-flop will not be used in a given application. Any unused inputs will always be omitted from the logic diagram. When pins C or E are not explicitly shown, it is implied that they are grounded. When any of the other inputs are not shown, it is implied that they are left open.

It should be mentioned that diodes D_1 and D_2 in Figure A. 3 may be omitted when the flip-flop is used with low input frequencies. These diodes are not needed if the input on J or K remains at the "one" level for at least 0. 25 microseconds before switching to "zero".

It is possible to add more AC inputs to the basic flip-flop by adding more resistor - capacitor-diode gates to the basic

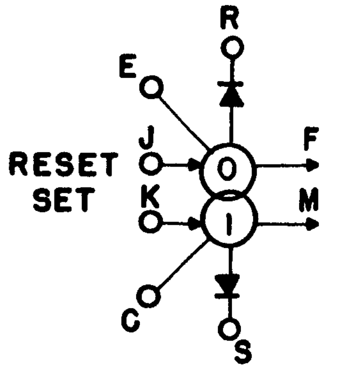


FIGURE A.5- AC SET-RESET FLIP-FLOP

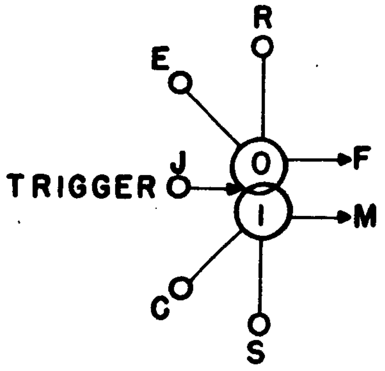


FIGURE A.6- TRIGGER FLIP-FLOP

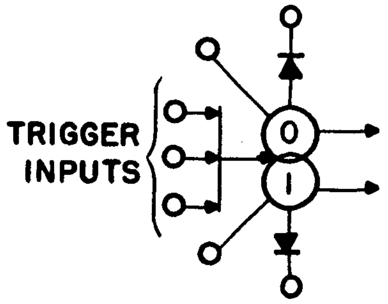
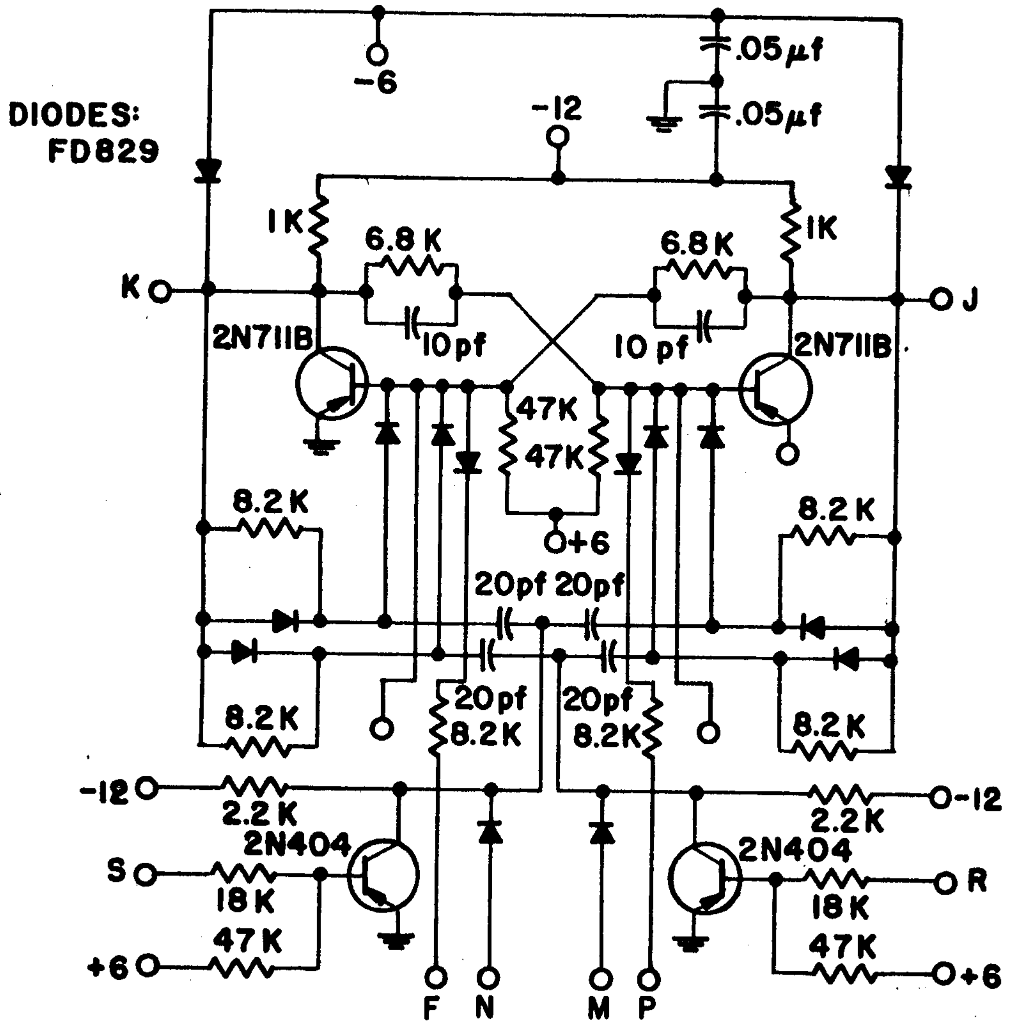


FIGURE A.7- MULTIPLE TRIGGER FLIP-FLOP

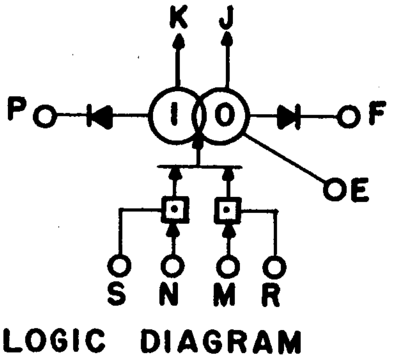
circuit. The flip-flop can then be switched by any one of a number of inputs. This type of logical "OR" was applied only in the binary counters, where flip-flops with up to three trigger inputs were employed. The notation for such a multiple trigger input flip-flop is shown in Figure A. 7.

The bidirectional counter circuit, shown in Figure A. 8, uses a two input trigger flip-flop. Terminals N and M are connected to the "one" and "zero" sides of the preceding counter stage. Levels S and R control whether input N or M triggers the flip-flop and hence control whether the counter counts backward or forward. When one of these levels is true, the corresponding trigger input is clamped at ground by a saturated transistor and inhibits this input from triggering the flip-flop. Changes in the control levels S and R do not trigger the flip-flop since the rise time of the 2N404 transistor is far too large. It is necessary to wait two microseconds after the control levels change before triggering the counter in order to allow the levels on the trigger inputs to stabilize.

One other circuit must be included here for completeness - the four megacycle crystal oscillator. The circuit is illustrated in Figure A. 9. The oscillator proper consists of transistor Q_1, with positive feedback from collector to base. Transistor Q_2 serves as an emitter follower to isolate the oscillator. It also provides a current source drive for the tunnel diode, TD-2, and



A. CIRCUIT DIAGRAM



B. LOGIC DIAGRAM

FIGURE A.8– 5 MC BIDIRECTIONAL COUNTER CIRCUIT AND LOGIC DIAGRAM

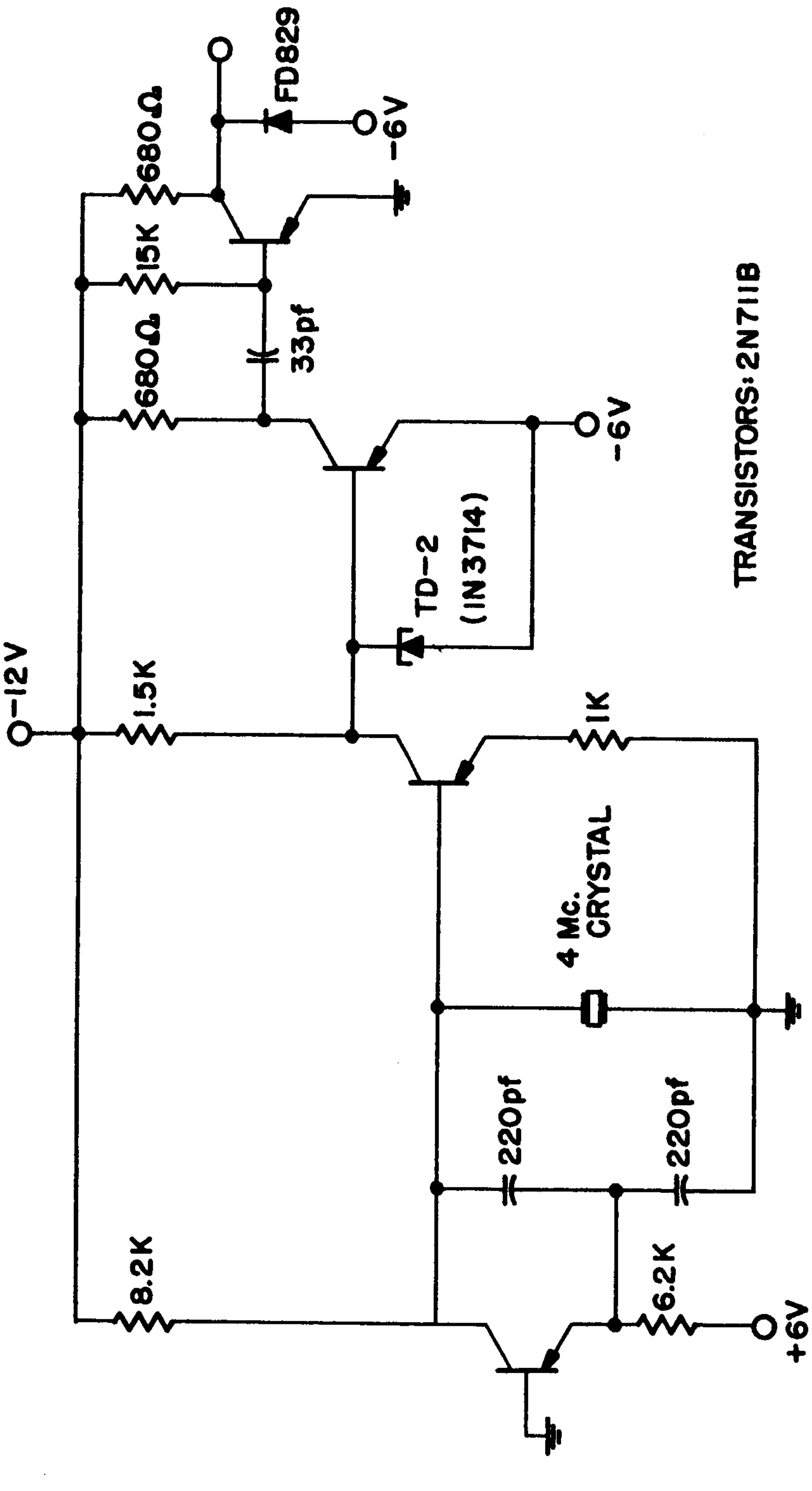


FIGURE A.9 — 4 Mc. CRYSTAL OSCILLATOR

clamps the collector of Q_1 at approximately -6 volts. The tunnel diode is used as a threshold element to square up the sinusoidal oscillator output. Transistor Q_3 amplifies the tunnel diode output. A pulse generator was used as the output stage not because pulses were required, but to obtain an output signal with the proper voltage levels.

APPENDIX B
DETAILED TIMING AND LOGIC

B. 1 Machine Operations

In order to obtain the Boolean functions realized by the various combinational circuits in Chapter Three and to understand the detailed timing, it is convenient to list all possible control functions in the order in which they occur in time, along with the operations they define. This list is shown in Table B. 1. The transfers specified on a given line are executed when the clock, C_M, switches from "one" to "zero" provided the corresponding control function is true.

The manner in which the output of the operational multiplier $K_i f$, is processed requires a special interpretation since it may emit as many as four distinct pulses between successive "zero" to "one" transitions of C_M. rA is incremented with the multiplier output as long as the indicated control function is true. The circuitry was designed with time delays so that rA is incremented if a pulse occurs at the multiplier output when the control level changes.

The reasons for most of the operations should be clear after reading Chapter Three. Some new signals and operations have been defined which will be clarified in the ensuing sections. These new signals and operations are associated, for the most

	CONTROL FUNCTION	OPERATIONS PERFORMED WHEN CONTROL FUNCTION IS TRUE
1	$S_0(B) \cdot P_{03}$	$S_A \longrightarrow W$
2	$S_0(B) \cdot S_1(M_R)$	$0 \longrightarrow rH$
3	$S_0(B) \cdot S_2(M_R)$	$rY \longrightarrow rH$
4	$S_0(B) \cdot \overline{P}_{03}\,\overline{A}_e S_A$	$rA_R + 1 \longrightarrow rA_R$; $1 \longrightarrow W$
5	$S_0(B) \cdot \overline{P}_{03}\,\overline{A}_e \overline{S}_A$	$rA_R - 1 \longrightarrow rA_R$; $1 \longrightarrow W$
6	$S_0(B) \cdot S_{127}(M_R)$	$S_1(B) \longrightarrow rB$; Inhibit $K_i f$
7	$S_1(B) \cdot S_1(M_R)$	$S_K \oplus R \longrightarrow S_p$; $R \longrightarrow W$
8	$S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot S_0(M_R) \cdot \overline{S}_A$	$+127 \longrightarrow rH$
9	$S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot S_0(M_R)\; S_A$	$-127 \longrightarrow rH$
10	$S_1(B) \cdot S_1(W)\,\overline{A}_e\, S_2(M_R)$	$0 \longrightarrow rA$
11	$S_1(B) \cdot \overline{P}_{04} S_p R$	Count rA backward with $K_i f$; $R \longrightarrow W$
12	$S_1(B) \cdot \overline{P}_{04} \overline{S}_p R$	Count rA forward with $K_i f$; $R \longrightarrow W$
13	$S_1(B) \cdot S_{127}(M_R)$	$S_2(B) \longrightarrow rB$; Inhibit $K_i f$
14	$S_2(B) \cdot S_1(M_R)$	$S_K \oplus R \longrightarrow S_p$; $R \longrightarrow W$
15	$S_2(B) \cdot \overline{P}_{04} S_p R$	Count rA backward with $K_i f$; $R \longrightarrow W$
16	$S_2(B) \cdot \overline{P}_{04} \overline{S}_p R$	Count rA forward with $K_i f$; $R \longrightarrow W$
17	$S_2(B) \cdot S_{127}(M_R)$	$S_3(B) \longrightarrow rB$; Inhibit $K_i f$
	. . . .	. . . .
18	$S_{15-M}(B) \cdot S_1(M_R) \cdot S_1(W)$	$S_E \oplus S_K \longrightarrow S_p$; $S_E \longrightarrow W$
19	$S_{15-M}(B) \cdot S_1(M_R) \cdot \overline{S_1(W)}$	$R \oplus S_K \longrightarrow S_p$; $R \longrightarrow W$
20	$S_{15-M}(B) \cdot \overline{P}_{04} \cdot S_1(W) \cdot \overline{E}_e S_p$	Count rA backward with $K_i f$; $1 \longrightarrow W$
21	$S_{15-M}(B) \cdot \overline{P}_{04} \cdot S_1(W) \cdot \overline{E}_e \overline{S}_p$	Count rA forward with $K_i f$; $1 \longrightarrow W$
22	$S_{15-M}(B) \cdot \overline{P}_{04} \cdot \overline{S_1(W)} \cdot R\, S_p$	Count rA backward with $K_i f$; $R \longrightarrow W$
23	$S_{15-M}(B) \cdot \overline{P}_{04} \cdot \overline{S_1(W)} \cdot R\, \overline{S}_p$	Count rA forward with $K_i f$; $R \longrightarrow W$
24	$S_{15-M}(B) \cdot S_{127}(M_R)$	$S_{16-M}(B) \longrightarrow rB$; Inhibit $K_i f$
	. . . .	. . . .
25	$S_{15}(B) \cdot S_1(M_R)$	$S_K \oplus R \longrightarrow S_p$; $R \longrightarrow W$
26	$S_{15}(B) \cdot \overline{P}_{04} S_p R$	Count rA backward with $K_i f$; $R \longrightarrow W$
27	$S_{15}(B) \cdot \overline{P}_{04} \overline{S}_p R$	Count rA forward with $K_i f$; $R \longrightarrow W$
28	$S_{15}(B) \cdot S_{127}(M_R) \cdot G$	$S_0(B) \longrightarrow rB$; $1 \longrightarrow rW$; Inhibit $K_i f$
29	$S_{15}(B) \cdot S_{127}(M_R) \cdot \overline{G}$	$S_1(B) \longrightarrow rB$; $rW + 1 \longrightarrow rW$; Inhibit $K_i f$

TABLE B.1– DETAILED SEQUENTIAL OPERATION OF THE COMPENSATOR

part, with the details of processing the sign bit, the gating of information into and out of the machine, and special operations to be performed in the event the compensator output saturates.

B.2 Control Logic

In this section, the sample time selection switch, the feedback gating for rB, the special sequential circuit, and the two registers in the control logic will be described in detail.

The sample time switch is wired to provide a three bit coded output to define when rW has accumulated the desired number of iterations of the basic computation cycle. The inputs to the first two stages of rB are generated from this information by the feedback gating.

The sample time programming switch is shown in Figure B. 1. The input variables $C_o(W)$, $C_1(W)$, $C_2(W)$ are the first, second, and third stages, respectively, of the binary counter, rW. Their negotiations are obtained from the zero sides of these flip-flops. It should be clear that the outputs of the switch will all be "zero" only if the count in rW equals the position number of the switch. The switch was wired so that all zeros would denote equality in order that the outputs could be ANDed with a NOR gate.

The input expressions, I_o and I_1, which define the inputs to cells $C_o(B)$ and $C_1(B)$, respectively, may be obtained from lines 6, 28, and 29 of Table B. 1:

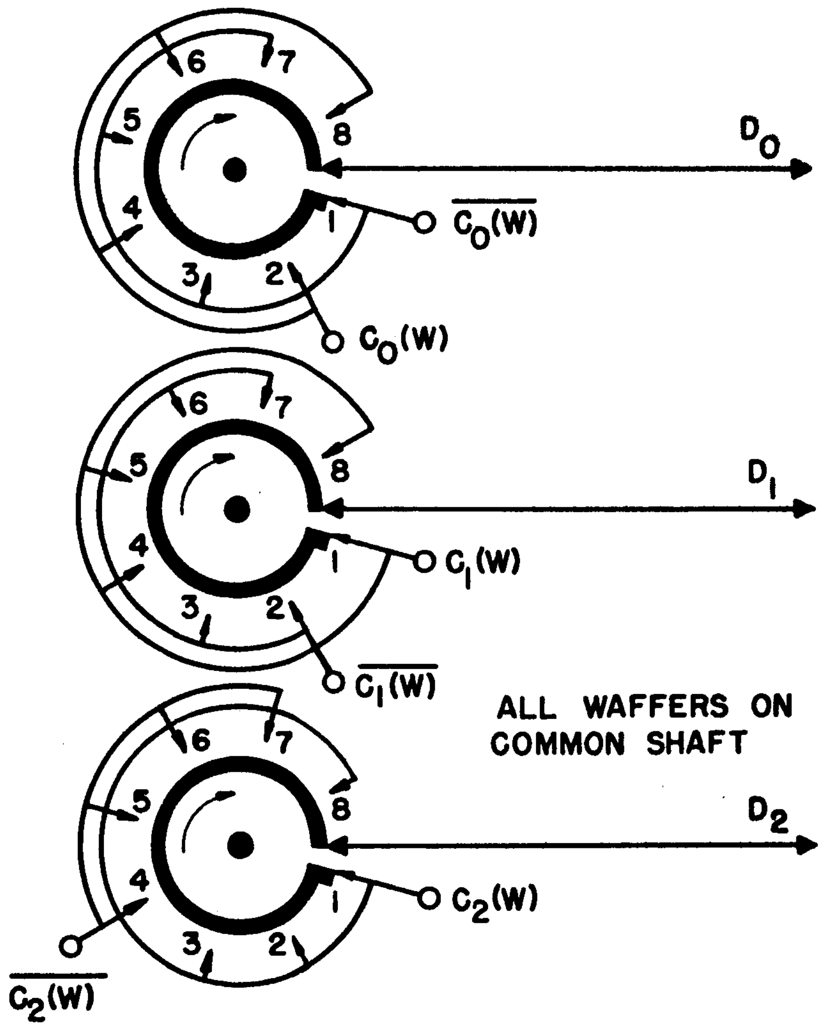


FIGURE B.1– SAMPLE TIME PROGRAMMING SWITCH

$$I_0 = S_{15}(B)\,G$$

$$I_1 = S_{15}(B)\,\overline{G} \cup S_0(B).$$

G is defined to equal "one" when the count in rW equals the sample time switch position number. Control signal $S_{127}(M_R)$ does not appear in the above expressions because it is generated by the overflow from rM which supplies the shift command.

Now note that

$$S_{15}(B)\,\overline{I}_0 = S_{15}(B)\,(\overline{S_{15}(B)} \cup \overline{G})$$

$$= S_{15}(B)\,\overline{G}\ .$$

Therefore $I_0 = S_{15}(B)\,G,$

and $I_1 = S_{15}(B)\,\overline{I}_0 \cup S_0(B)\ .$

If we let D_0, D_1, D_2 denote the three outputs from the sample time switch, we obtain

$$I_0 = S_{15}(B)\ \overline{D}_0\,\overline{D}_1\,\overline{D}_2\ ,$$

and $I_1 = S_{15}(B)\ \overline{I}_0 \cup S_0(B)\ .$

The complements of I_0 and I_1 are also required for inputs to the shift register, and so the circuit was instrumented as shown in Figure B. 2.

The control signals P_{03} and P_{04} are defined as follows:

$$P_{03} = S_0(M_R) \cup S_1(M_R) \cup S_2(M_R)$$

and $P_{04} = S_0(M_R) \cup S_1(M_R) \cup S_2(M_R) \cup S_3(M_R).$

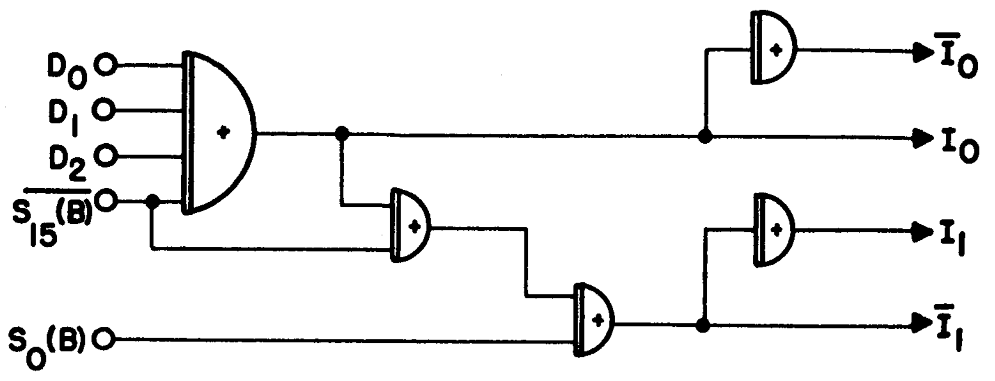


FIGURE B.2– rB FEEDBACK GATING

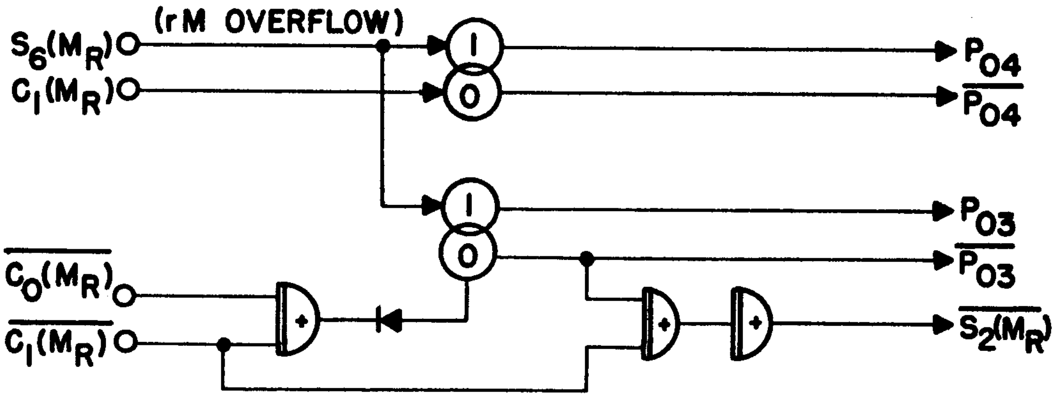


FIGURE B.3– SEQUENTIAL CIRCUIT FOR REALIZING MISCELLANEOUS CONTROL SIGNALS

In other words, P_{03} is a signal which switches to "one" when rM clears to zero and remains at "one" for three microseconds; P_{04} switches to "one" when rM clears and remains at "one" for four microseconds. The use of these signals will become clear in the following sections.

The above functions, as well as $\overline{S_2(M_R)}$, were realized with a sequential circuit to save combinational gating and to avoid the hazards present in a combinational circuit.

The sequential circuit is shown in Figure B. 3. Both flip-flops are set when rM overflows, i. e. , when $S_0(M_R) \rightarrow rM_R$. The fact that these flip-flops are reset at the appropriate time to realize P_{03} and P_{04} may be easily verified.

The control logic is now shown in detail in Figure B. 4. The implementation of the registers should be clear from the description of the circuits in Appendix A. The four megacycle master oscillator is gated with two NOR gates connected as a set-reset flip-flop in order to obtain reliable operation in the event of contact bounce when the stop-run switch is thrown.

B. 3 Programmable Diode Matrix

The programmable diode matrix that was used is manufactured by AMP, Incorporated. It realizes the circuit shown in Figure B. 5. The resistors and output diodes are mounted external to the matrix. The programming diodes are mounted in

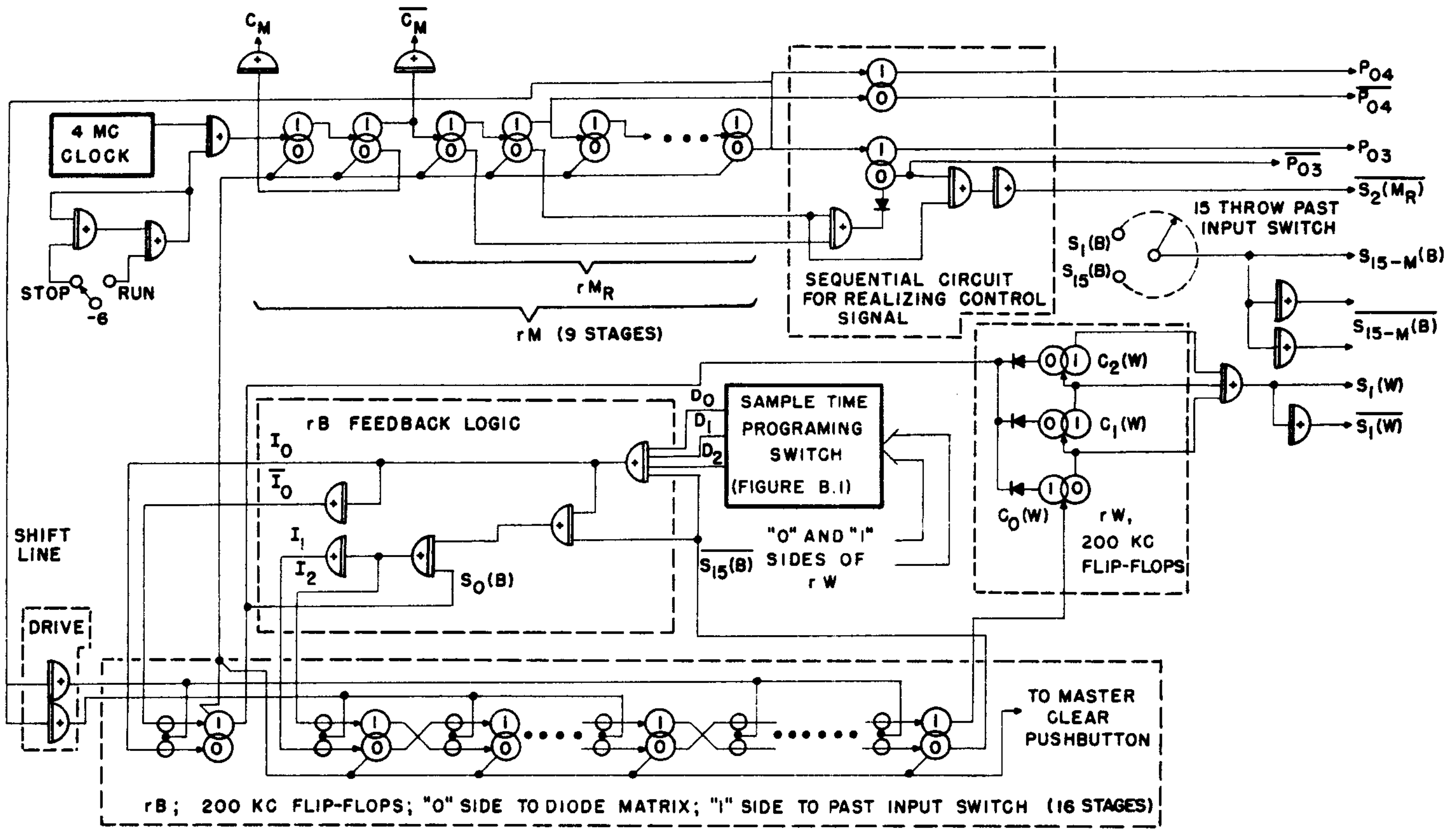


FIGURE B.4— DETAILED CONTROL LOGIC

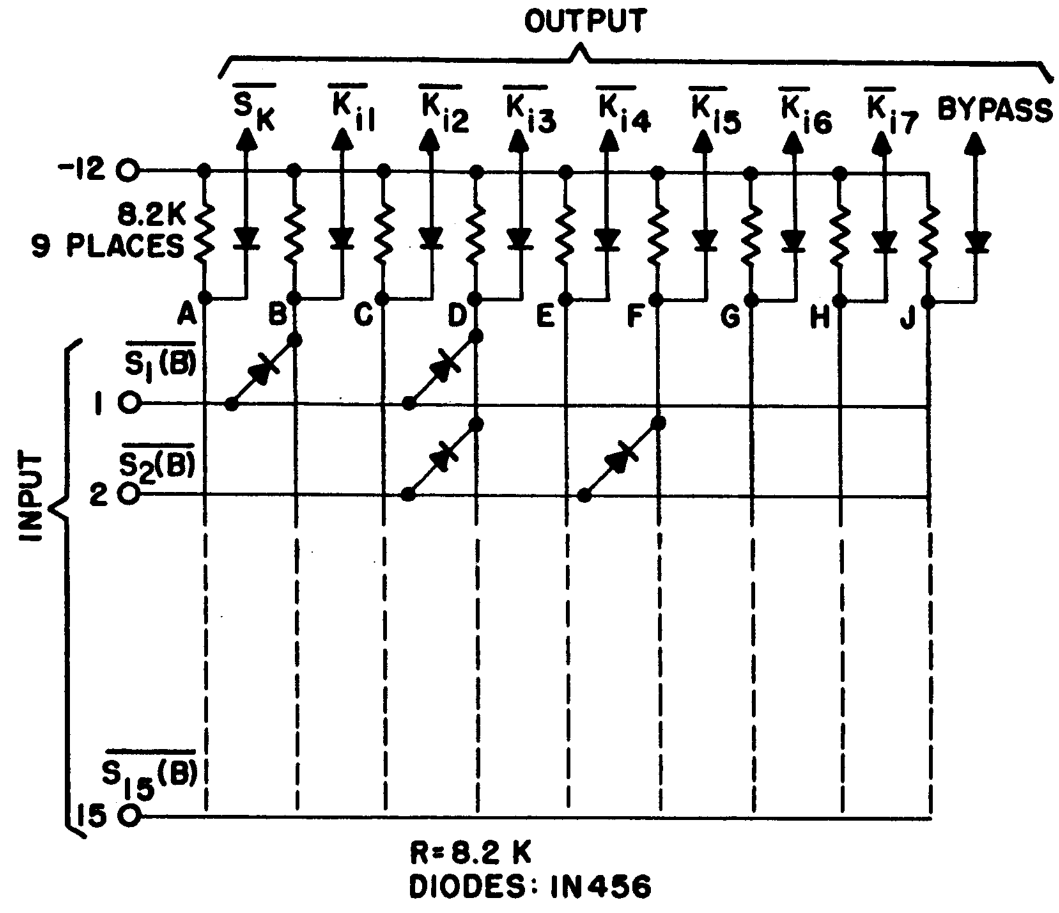


FIGURE B.5— PROGRAMMABLE DIODE MATRIX CIRCUIT

pins so that the location of diodes in the matrix can be conveniently changed by removing or inserting these pins.

The numbered rows were used as inputs and were driven by rB. Therefore, only outputs were taken from the lettered columns. A given output will be "zero" only if a pin is present at the intersection of this output column with the input row that is at zero volts. Since the outputs are defined to represent the complements of the indicated variables, a binary "one" is represented in the matrix by the presence of a diode pin.

The diodes that are in series with the output (called output diodes above) are present to offset the voltage drop across the diodes in the matrix. This ensures that the output will be adequate to turn off a transistor.

The physical appearance of the diode matrix may be clearly seen in the photograph of the compensator, Figure 1.4.

B.4 Operational Multiplier Gating

The operational multiplier gating is illustrated in Figure B.6. Recall that f is the input to rM and $C_0(M), \ldots, C_6(M)$ are the seven initial stages of rM. $K_{i1}, \ldots, K_{i7}$ are the seven digits which specify the magnitude of the weighting constant, K_i. K_{i1} carries a weight of 2^{-1}, K_{i2} carries a weight of 2^{-2}, etc. The diode matrix (see previous section) supplies the signals, $\overline{K}_{i1}, \ldots, \overline{K}_{i7}$ directly, with no need for inversion. BYPASS is a signal which is also programmed into the diode matrix. A

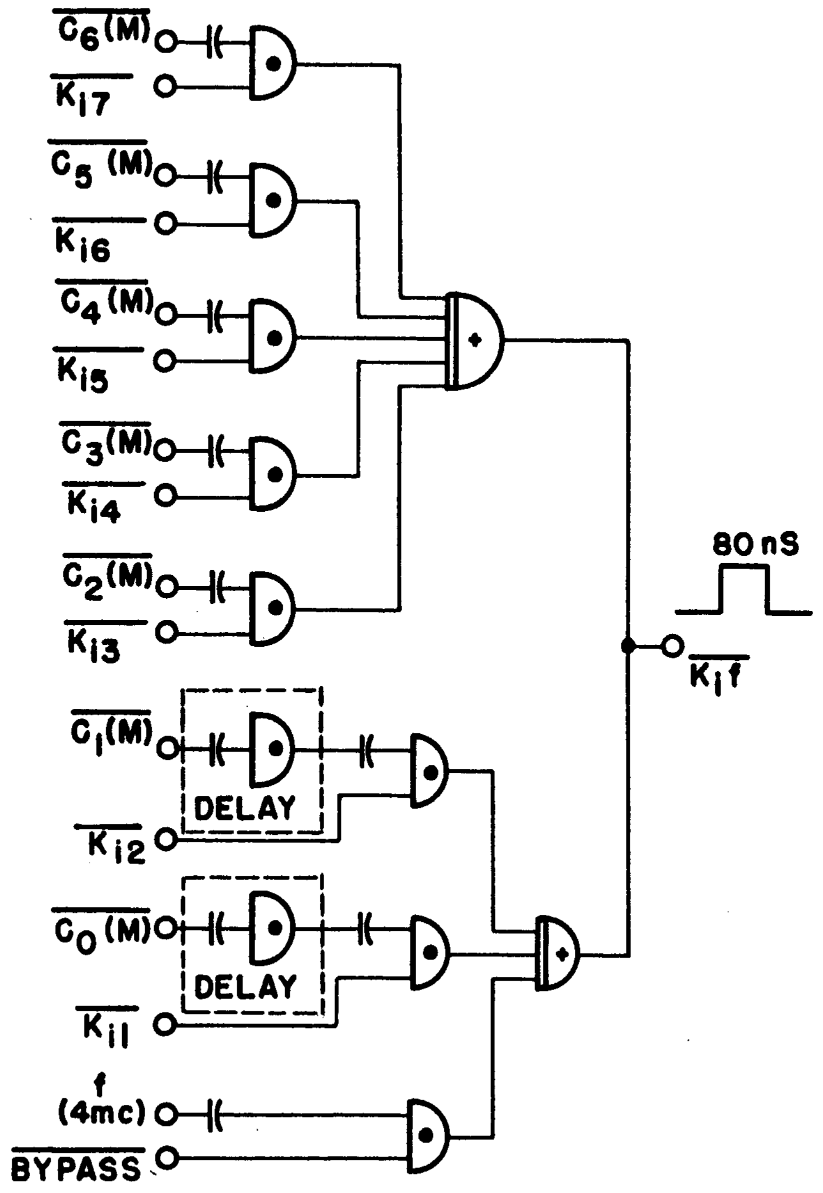


FIGURE B.6– OPERATIONAL MULTIPLIER GATING

programming constraint which must be obeyed is that BYPASS can be programmed to equal "one" only if all the variables $K_{i1}, \ldots, K_{i7}$ are programmed to equal "zero".

The output is obviously given by:

$$K_i f = \bigcup_{j=1}^{7} K_{ij} \, P_{C_{j-1}(M)} \quad \cup \; (\text{BYPASS}) \quad P_f . \qquad (\text{B. 1})$$

(P_x is a pulse which occurs when x switches from "zero" to "one" where x is any Boolean variable.) If BYPASS = 1, $K_{ij} = 0$ by definition, and therefore $K_i f = P_f$; the output frequency equals the input frequency f. (To save notation, the same symbol is used for a Boolean variable and its frequency in this section.) When BYPASS = 0, the output is:

$$K_i f = \bigcup_{j=1}^{7} K_{ij} \, P_{C_{j-1}(M)} .$$

Since $C_{j-1}(M)$ is the (j-1)st stage of a binary counter which is incremented with an input signal of frequency f, $P_{C_{j-1}(M)}$ is of frequency $f/2^j$. Furthermore, since only one stage of a binary counter switches from "zero" to "one" each time it is incremented, all the pulses in the signals $P_{C_{j-1}(M)}$ are distinct in time; i.e., the Boolean variables $P_{C_{j-1}(M)}$ are mutually disjoint. The output frequency is therefore:

$$\sum_{j=1}^{7} K_{ij} \frac{f}{2^j} = f \sum_{j=1}^{7} K_{ij} \cdot 2^{-j} ,$$

where K_{ij} is now viewed as a real number equal to zero or one. This last summation is just the definition of the constant K_i. The output frequency is therefore $K_i f$.

The delay in two of the input legs is required to keep the pulses from these two inputs from running together with the pulse from the last input (input $\overline{C_6(M)}$). Since the inputs come from a transition coupled binary counter, $C_6(M)$ will change about 7(25) = 175 nanoseconds after the counter receives an input while $C_0(M)$ changes about 25 nanoseconds after receipt of an input. Since the output pulses are eighty nanoseconds wide, the pulses would not be sufficiently separated in time to be distinguished by a counter without the added delay elements.

B. 5 Processing the Sign Bit

The sign of the product of a weighting constant and past input or output is held in a flip-flop, S_P. The input to this flip-flop, I_{SP}, defines the sign of the product at the instant it is clocked into S_P.

Each time rB changes state, a new constant is gated into the operational multiplier and a new value for the sign of the product must be transferred into S_P. The transfer is executed by $\overline{S_2(M_R)}$, a control signal which changes to "zero" two microseconds after the clock transition which causes rB to change state. The two microsecond delay is sufficient for rB to change state, for the sign bit output of the diode matrix to settle, and for

I_{SP} to stabilize. Two additional microseconds must then be allowed for the forward and backward command levels within the bidirectional counter to stabilize before the operational multiplier output is gated into rA.

The sign bit, therefore, is not actually located in the first bit position of each word. The first bit position is not used; the second position holds the sign bit, and the third position is also blank. The fourth position contains the first bit of the magnitude of the word. The sign was actually written into the first three positions, however, so that the control signal P_{03} (see Section B. 3) could be used to define the times at which the sign was to be written.

The expression for I_{SP} may be obtained by inspecting those lines in Table B. 1 which define a transfer into S_P. We obtain:

$$I_{SP} = (S_K \oplus R) \cdot \overline{S_{15-M}(B)} \;\cup\; (S_K \oplus R) \cdot S_{15-M}(B) \cdot \overline{S_1(W)} \;\cup\; (S_E \oplus S_K) \cdot S_{15-M}(B) \cdot S_1(W). \qquad \text{(B.2)}$$

The control signal $S_1(M_R)$ does not appear in the expression because this ANDing operation is performed by clocking I_{SP} into S_P when $S_1(M_R) = 1$.

Equation B. 2 may be decomposed into two equations:

$$I_{SP} = S_K \oplus X \qquad \text{(B. 3a)}$$

where $X = R\,(\overline{S_1(W)}\ \cup\ \overline{S_{15-M}(B)})\ \cup S_E \cdot S_1(W) \cdot S_{15-M}(B).$

(B. 3b)

These equations were instrumented as shown in Figure B. 7. The ring sum was instrumented as suggested in reference 16 since the asserted forms of S_K and X are not required for this instrumentation.

B. 6 Accumulator Design

The accumulator is a two's complement bidirectional counter with positive numbers represented by "zero" in the sign bit. Two's complement representation for negative numbers was selected since this form yields the simplest counter and is ideal for direct conversion to analog form (see Appendix D). The counter accepts a single count input and has two inputs, F_A and B_A, which define whether the counter should count forward or backward.

The number of bits in rA was selected so that it could never overflow at any time during a computation cycle. The maximum number of counts rA will ever have to accept is:

$$[\,3 \times 127 + (127 - 15)] \times 15 \times 8 = 59,160.$$

The number in brackets equals the maximum number of counts that rA must accept while accumulating a single product. Up to fifteen products may be formed in a single iteration of the computation cycle, and there are up to eight iterations. The product

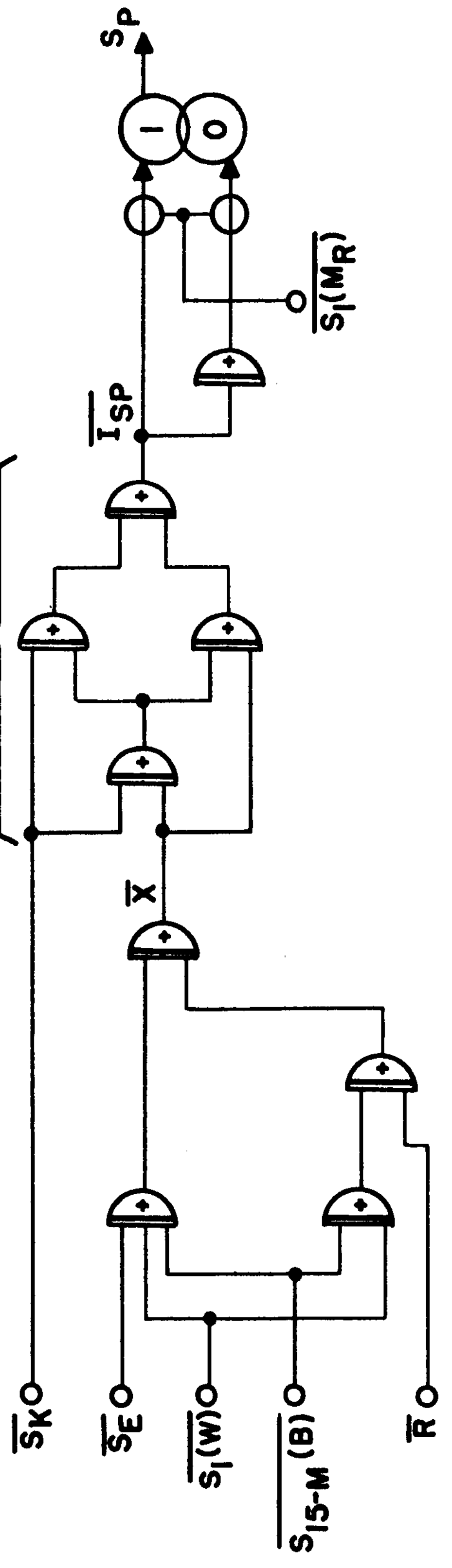


FIGURE B.7— SIGN OF PRODUCT GATING

of these numbers, therefore, equals the maximum count. It follows that seventeen bit counter was needed, including one bit for the sign.

The accumulator is shown in Figure B. 8. The small resistors in series with the flip-flop outputs are necessary to isolate the flip-flops from the stray capacitance associated with cabling and the gain switch.

The extra trigger inputs on the first six stages are used for the count input to rA_R, the counter defined by those stages of rA which are used for the output and the stages of rA which lie to the right of the output stages. It is rA_R which must be counted toward zero in order to insert y_n into the delay line, not rA. The gain switch is used to select the proper trigger input, as well as the proper output stages.

B. 7 rA Trigger Input Gating

The trigger input to rA is obtained by examining those lines in Table B. 1 which define when rA is to be incremented with the operational multiplier output, $K_i f$. As explained in Section B. 5, the input should remain blocked until four microseconds after rB enters a new state. The control signal P_{04} is used to perform this function. The input is:

$$\begin{aligned}\overline{I}_A = \overline{P}_{04} \quad [\,&\overline{S_0(B)} \cdot \overline{S_{15-M}(B)} \cdot (K_i f) \cdot R \quad U \\ &S_{15-M}(B) \cdot S_1(W) \cdot \overline{E_e} \cdot (K_i f) \quad U \\ &S_{15-M}(B) \cdot \overline{S_1(W)} \cdot R \cdot (K_i f)]\end{aligned}$$

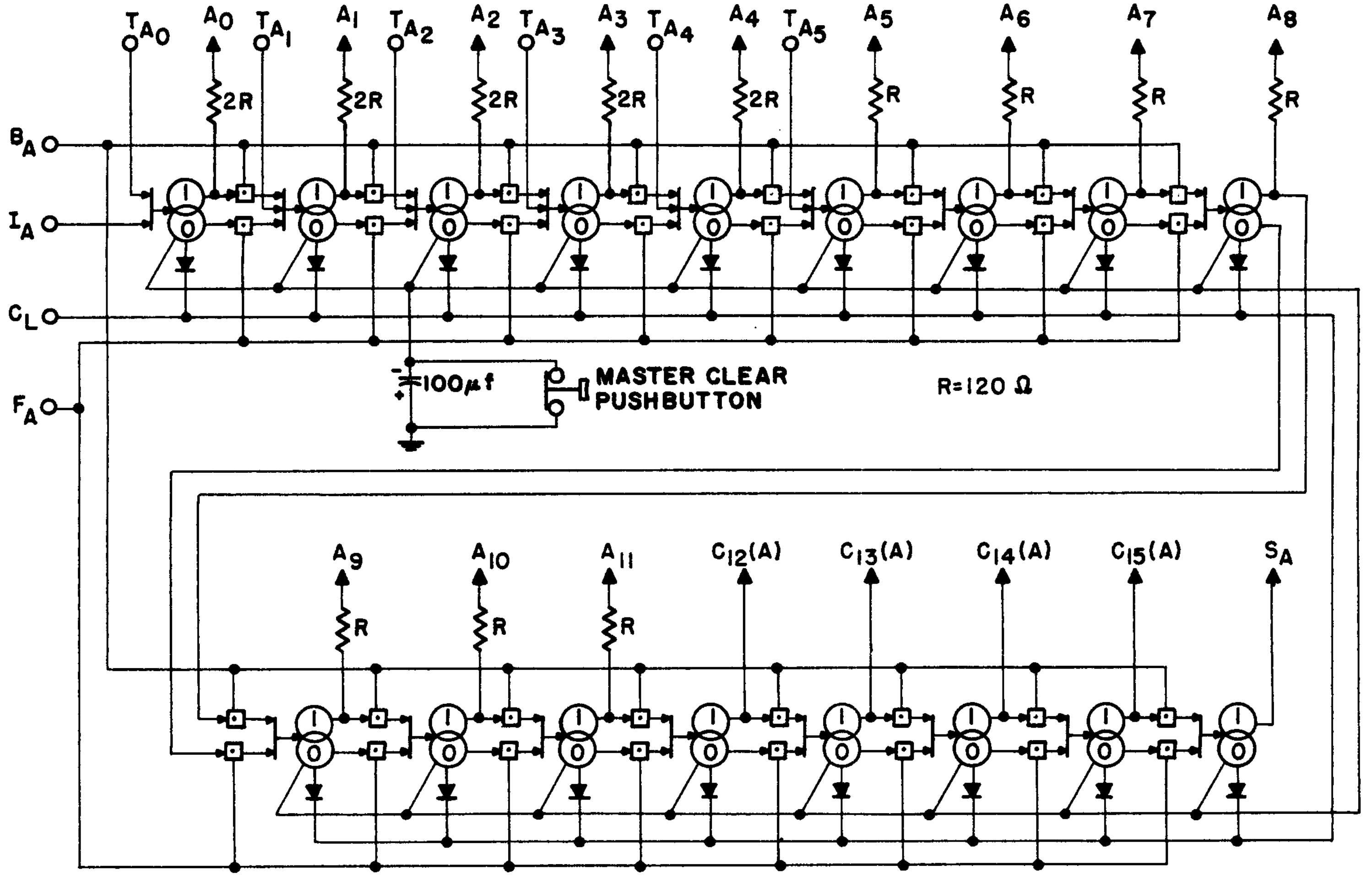


FIGURE B.8- BIDIRECTIONAL COUNTER, rA

We write $\overline{T}_A$ since we wish to have an expression which is normally at the "one" level and rises to "zero" when $K_i f = 1$.

The expression was instrumented as shown in Figure B. 9. $K_i f$ was not factored out of the expression so that a pulse from the operational multiplier would have to probagate through two additional gates. This helps insure that the control level P_{04} will set up the gating before the output of the multiplier influences the output of the circuit.

Note that E_e and R are clocked into flip-flop's before they are applied to the combinational gating. This was done since the output of this gating must be defined by their value at the time a clock transition occurs, not the value they change to. When C_M switches to "zero", E_e and R are transferred into the flip-flops and set up the gating before any pulse from the multiplier arrives. In other words, the delay through the operational multiplier gating was designed to be longer than the time required to transfer E_e and R into the flip-flops.

B. 8 Directional Gating for rA

The count forward (F_A) and count backward (B_A) commands to rA are defined not only by S_P, but also by S_A, the sign of rA. These levels are defined by S_A while y_n is being written into memory. We conclude from Table B. 1 that

$$F_A = S_0(B) \cdot S_A \cup \overline{S_0(B)} \cdot \overline{S_P}$$

and

$$B_A = \overline{F_A} \quad .$$

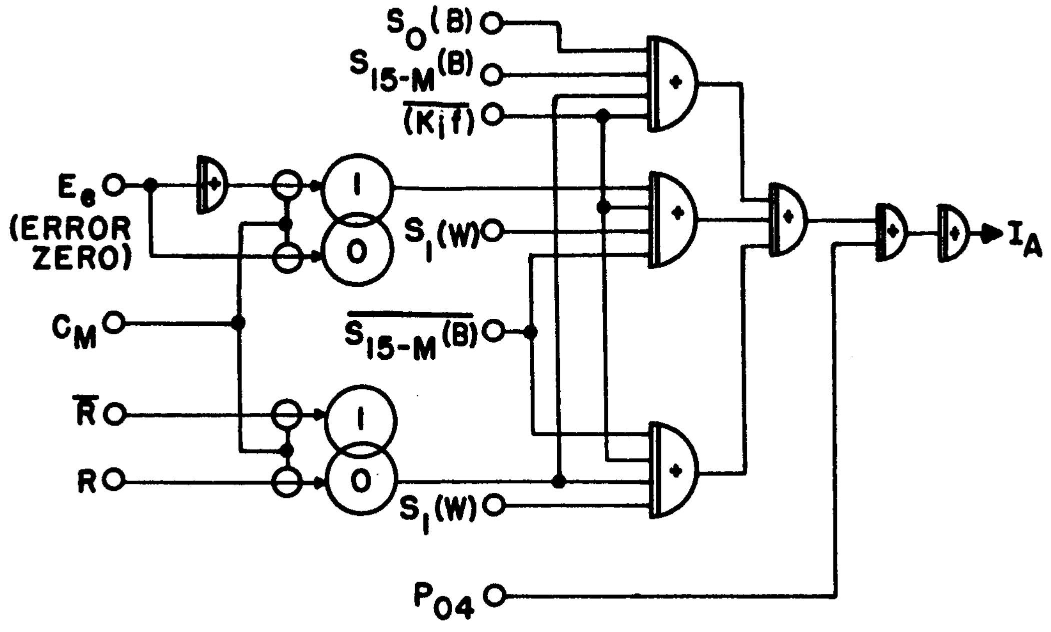


FIGURE B.9– rA INPUT GATING

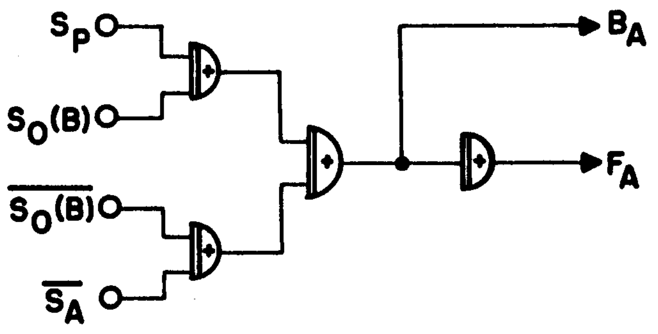


FIGURE B.10– DIRECTIONAL GATING FOR rA

The instrumentation is shown in Figure B. 10.

B. 9 Gain Programming Switch

The function of the gain programming switch is threefold:

1. It selects the stages of rA that will be used for the output, y_n.
2. It selects the proper trigger input for the master clock which counts rA_R toward zero as y_n is inserted into memory.
3. It selects the stages of rA which will serve as inputs to the gating that forms the function A_e (rA_R empty).

The switch is shown in Figure B. 11. The inputs $A_0, \ldots, A_{11}$ are taken from the first twelve stages of rA through small isolation resistors. The outputs $C_0(Y), \ldots, C_6(Y)$, define the seven bit magnitude of y_n. The outputs $T_{A_0}, \ldots, T_{A_5}$, are the trigger inputs to the first 6 stages of rA. The outputs $F_1, \ldots, F_5$ are inputs to the combinational circuit that generates A_e.

B. 10 Combinational Circuit for A_e

A_e is defined to be true whenever the count in rA_R is zero. Obviously, rA_R is not zero if any cell in rA_R contains a "one". Therefore,

$$\bar{A}_e = \left[\bigcup_{i=K-1}^{15} C_i\,(A) \right] \cup S_A,$$

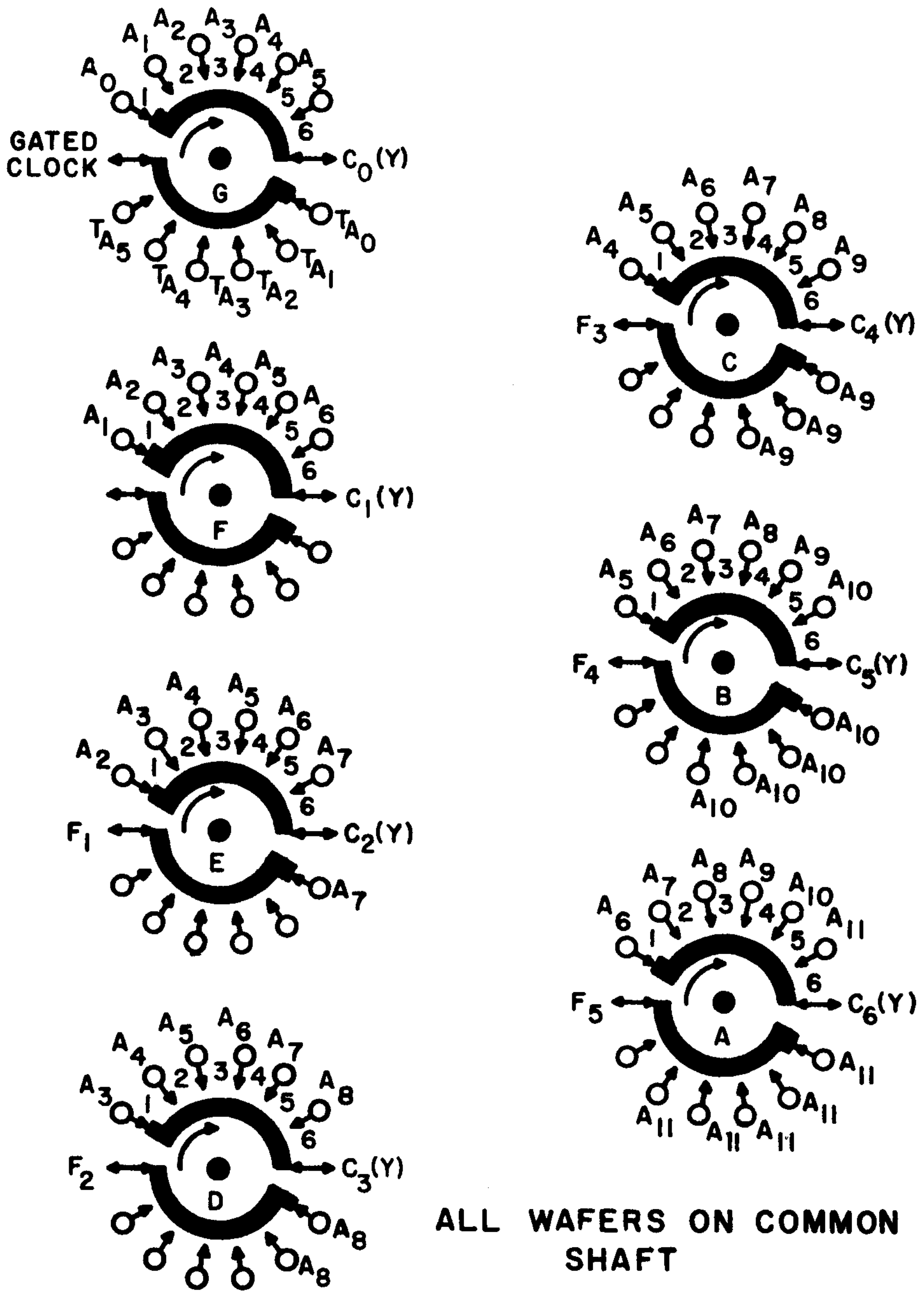

GATED CLOCK
$C_0(Y)$
G
A_0 A_1 A_2 A_3 A_4 A_5
T_{A_0} T_{A_1} T_{A_2} T_{A_3} T_{A_4} T_{A_5}
F
$C_1(Y)$
E
F_1
$C_2(Y)$
D
F_2
$C_3(Y)$
C
F_3
$C_4(Y)$
B
F_4
$C_5(Y)$
A
F_5
$C_6(Y)$
ALL WAFERS ON COMMON SHAFT

FIGURE B.11– GAIN PROGRAMMING SWITCH

where K is the gain setting. Since the first seven stages of rY are equal to the first seven stages of rA_R, we may write the above expression as:

$$\bar{A}_e = \left[\bigcup_{i=0}^{6} C_i\,(Y) \right] \cup \left[\bigcup_{i=6+K}^{15} C_i\,(A) \right] \cup S_A \;.$$

Since $C_i(A)$ is logically equivalent to A_i for $i=0, 1, \ldots, 11$, examination of Figure B. 11 will show that

$$\bigcup_{i=6+K}^{11} C_i(A) = \bigcup_{i=1}^{5} F_i \;.$$

(An open circuit corresponds to binary "zero" on a gate input.) We therefore have

$$\bar{A}_e = \left[\bigcup_{i=0}^{6} C_i\,(Y) \right] \cup \left[\bigcup_{i=1}^{5} F_i \right] \cup \left[\bigcup_{i=12}^{15} C_i\,(A) \right] \cup S_A .$$

The circuit is illustrated in Figure B. 12.

B. 11 Saturation Gating

In the event the compensator saturates, the compensator should supply the maximum output to the hold circuit. Also, it was decided that the compensator should supply a visual indication if saturation ever occurs.

In addition to deriving the logical expressions for performing the above functions, the equation for clearing rA before the start of a new computation will be derived in this section. Some of the gating may then be shared between these expressions.

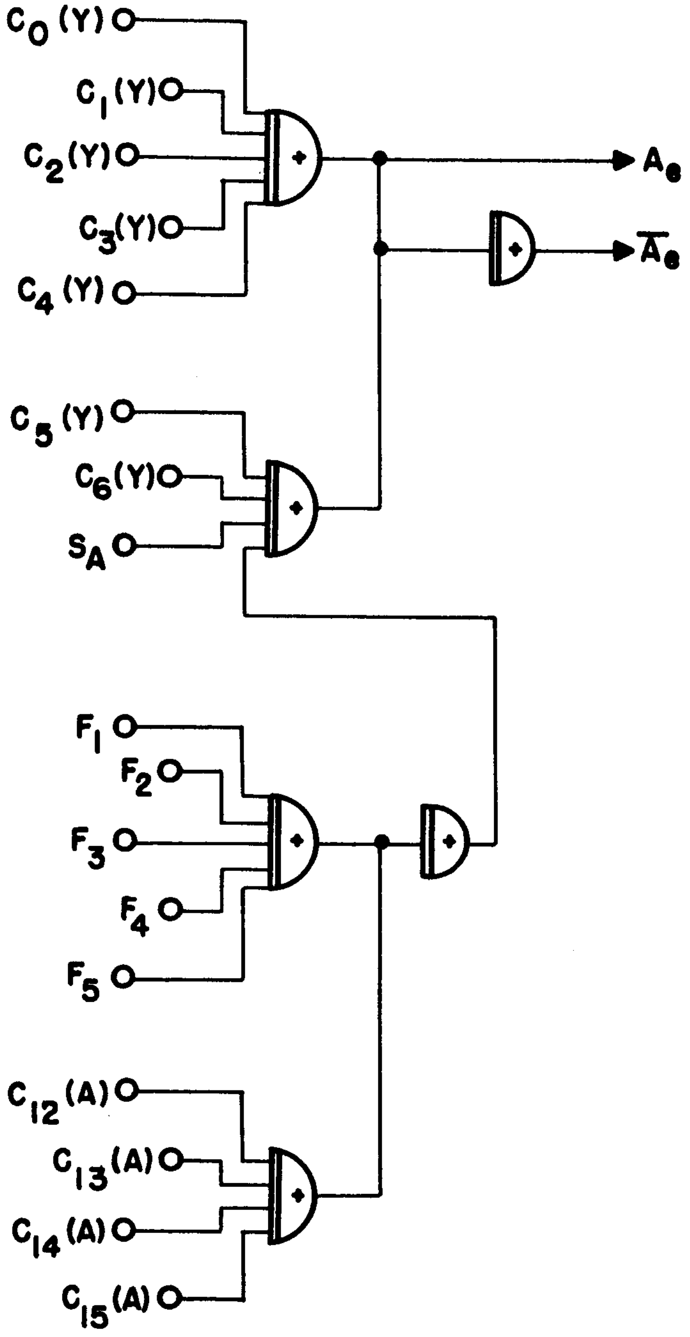


FIGURE B.12– COMBINATIONAL GATING FOR A_e

Ideally, if the compensator is saturated, it should transfer the maximum output to rH at the same time it would normally execute the transfer, i. e., at line 3 of Table B. 1. This, however, would require an additional combinational circuit on the output of rA to detect saturation. Also, the proper stages of rA would have to be switched to the input of this circuit. All of this gating can be eliminated by waiting until rA_R has been read into memory to find out if saturation has occurred. If rA_R is not zero after having been counted down by the delay line clock, we know that the compensator has saturated. In other words, we can use A_e to define when rA_R is zero and when the compensator is saturated. The only penalty we must accept is that the servo is driven in the proper direction by less than the maximum signal for not more than 1/16 of the time.

Saturation is therefore handled in the following manner:

1. Transfer rY to rH when rB enters state $S_0(B)$ whether or not saturation has occurred (line 3 of Table B. 1).
2. If A_e is still "zero" when rB enters state $S_1(B)$, then change the contents of rH to its maximum value (lines 8 to 9 of Table B. 1).

The DC set and reset inputs of the flip-flops in rH are used to execute step 2. Lines 8 and 9 of Table B. 1 show the set (S_H) and reset (R_H) inputs are defined by:

$$S_H = S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot P_{03} \cdot C_0(M_R) \cdot \overline{S}_A$$

and

$$R_H = S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot P_{03} \cdot C_0(M_R) \cdot S_A .$$

where $P_{03}\ C_0\ (M_R)$ is a one microsecond pulse which is switched to "one" by C_M when rM_R is in state $S_0\ (M_R)$. The set and reset pulses are, therefore, one microsecond wide. It is tempting to omit $C_0(M_R)$ and only use P_{03}, a three microsecond pulse. If this is done, however, false set and reset signals will occur because P_{03} changes before rB changes state.

An indicating flip-flop was used to remember that saturation has occurred; its input is:

$$R_H \cup S_H = S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot P_{03} \cdot C_0(M_R).$$

Finally, rA must be cleared at the start of each new computation cycle after rH has been adjusted (line 10 of Table B. 1). The clear command, C_L, is :

$$C_L = S_1(B) \cdot S_1(W) \cdot \overline{A}_e \cdot \overline{P}_{03} \cdot P_{04} ,$$

where we have used the fact that $\overline{P}_{03}\ P_{04} = S_3\ (M_R)$. The circuits were instrumented as shown in Figure B. 13.

B. 12 Serial Memory

The serial memory is a magnetostrictive delay line manufactured by the Computer Control Company, Inc. All read-write circuitry is included with the memory. The input is a combinational circuit consisting of four expandable three-input AND gates which feed an OR gate. An extra input to the OR gate is also provided for expanding the input circuit. The output of the OR gate

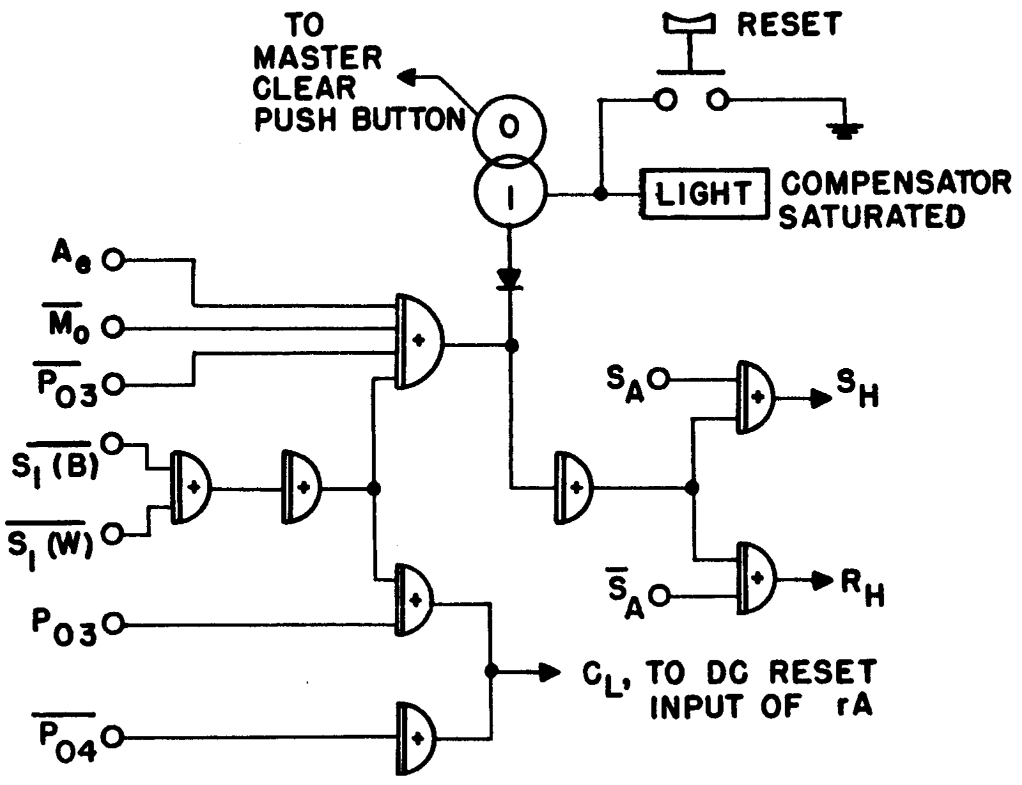


FIGURE B.13– SATURATION LOGIC AND rA CLEAR LOGIC

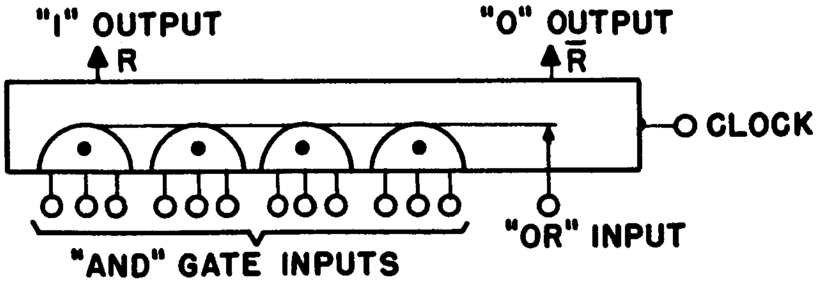


FIGURE B.14– SERIAL MEMORY LOGIC DIAGRAM

(which we shall call W) is clocked into the delay line.

The output of the delay line is a flip-flop which is set by the clock if the output of the delay line is "one"; the flip-flop is reset if the output is "zero". The memory is, therefore, logically equivalent to a large shift register.

Input and output logic levels are zero and -6 volts. The memory is clocked at one megacycle and has a delay time of 1920 microseconds. Therefore, it can store 1920 bits. The logic symbol suggested by Computer Control Company for the memory is shown in Figure B. 14.

B. 13 Memory Input Gating

The input to the write amplifier, W, may be read from Table B. 1:

$$\begin{aligned} W = {} & P_{03} [S_0(B) \cdot S_A \cup S_{15-M}(B) \cdot S_1(W) \cdot S_E] \cup \\ & \overline{P}_{03} \cdot S_0(B) \cdot \overline{A}_e \cup \overline{S_0(B)} \cdot \overline{S_{15-M}(B)} \cdot R \cup \\ & \overline{P}_{03} \cdot S_{15-M}(B) \cdot S_1(W) \cdot \overline{E}_e \cup S_{15-M}(B) \cdot \overline{S_1(W)} \cdot R. \end{aligned}$$

The input gating supplied with the memory unit may be used to realize most of this function. Only the function W_1 given by:

$$\begin{aligned} W_1 = {} & P_{03} \cdot S_{15-M}(B) \cdot S_1(W) \cdot S_E \cup \\ & \overline{P}_{03} \cdot S_{15-M}(B) \cdot S_1(W) \cdot \overline{E}_e \end{aligned}$$

had to be instrumented. W_1 was factored to give

$$W_1 = S_{15-M}(B) \cdot S_1(W) \cdot [P_{03} S_E \cup \overline{P}_{03} E_e] .$$

The gating is shown in Figure B. 15.

B. 14 Input - Output Timing Signals

The compensator must supply three signals to the analog to digital converter (see Appendix E):

1. The complement of the master clock which is used to count the error toward zero as it is read into memory.
2. A signal $\overline{R}_E$ which controls when the error is to be read.
3. A track command level, T_L, which controls when the converter is to track the analog error input.

The restrictions on the two timing signals are explained in Appendix E. When $\overline{R}_E$ switches to "zero" it opens a gate which allows the master clock, C_M, to count rE toward zero. Since rE is to be incremented by C_M only when the magnitude of the error is being loaded into memory, we have:

$$R_E = S_{15-M}(B) \cdot S_1(W) \cdot \overline{P}_{03} .$$

When switching from the track mode to the read mode, T_L must change at least two microseconds before R_E changes. A control signal which satisfies this requirement is $S_{15-M}(B)$. We can therefore identify T_L with $\overline{S_{15-M}(B)}$:

$$T_L = \overline{S_{15-M}(B)} .$$

In other words, the converter will continuously track the error except when rB enters state $S_{15-M}(B)$.

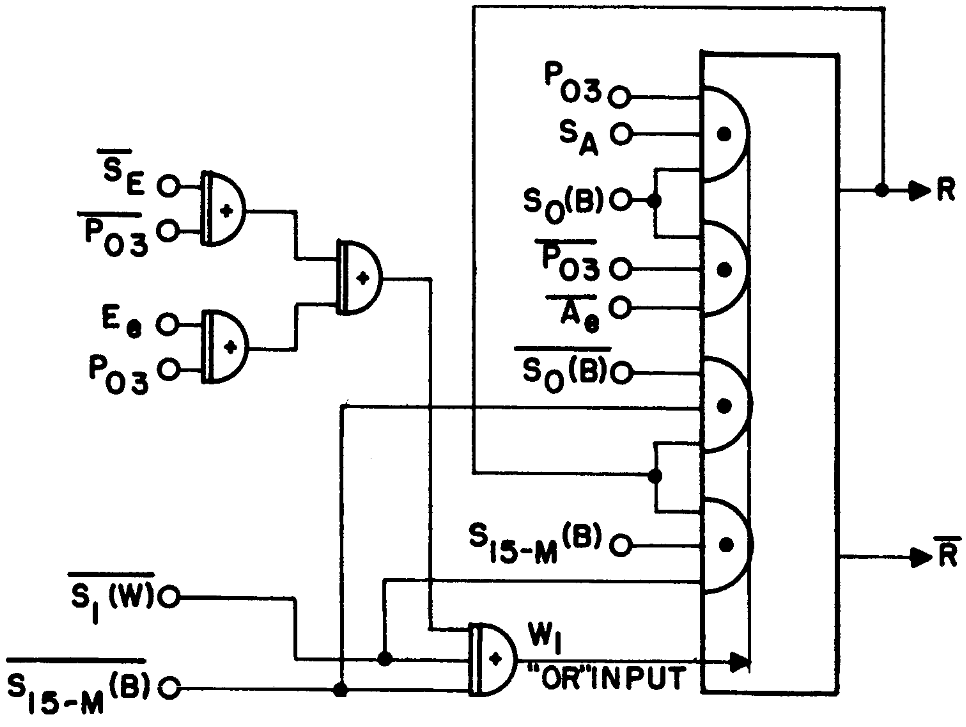


FIGURE B.15— MEMORY INPUT GATING

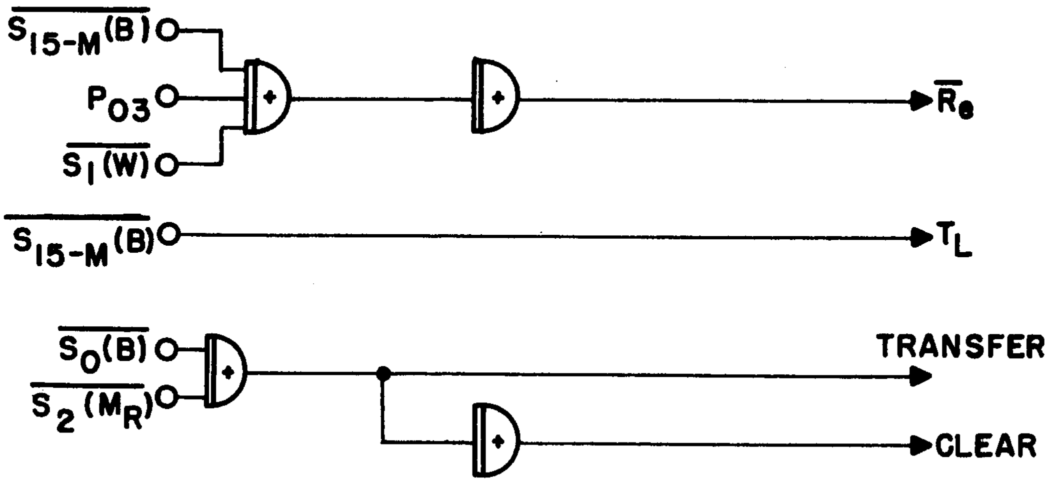


FIGURE B.16— INPUT-OUTPUT TIMING SIGNALS

A transfer into the zero-order hold is performed by first clearing the hold register, rH, and then executing the transfer. "One" to "zero" transitions are used to perform both of these operations. It should be clear from lines 2 and 3 of Table B. 1 that the following signals will supply the required transitions:

$$\text{TRANSFER} = S_0(B) \cdot S_2(M_R)$$

$$\text{CLEAR} = \overline{\text{TRANSFER}} \ .$$

The logic diagrams are shown in Figure B. 16.

APPENDIX C
ANALOG SIMULATOR

The analog simulator merely consists of five operational amplifiers which have their inputs and outputs brought out to five-way binding posts. The analog dynamics of the closed loop is simulated in real time by closing the amplifier loops in the appropriate manner. No provisions were made for setting in initial conditions since the response of the control loop to various initial conditions was of no interest. A stable closed loop will automatically drive all variables to zero and then the test inputs can be applied.

The operational amplifier circuit is shown in Figure C.1. The open loop gain of the amplifier is well in excess of 1,000 and is fairly independent of load. The gain-bandwidth product when operated with a 10K or smaller input resistor is approximately 500 KC.

The differential input stage (transistors Q_1 and Q_2) are biased at 40 microamperes. This results in a nominal input base current of 0.5 microamperes. This current must either be supplied by the driving source or by an external biasing network at the base of transistor Q_1.

The input stage is followed by two grounded emitter stages and an emitter follower. The emitter follower is biased

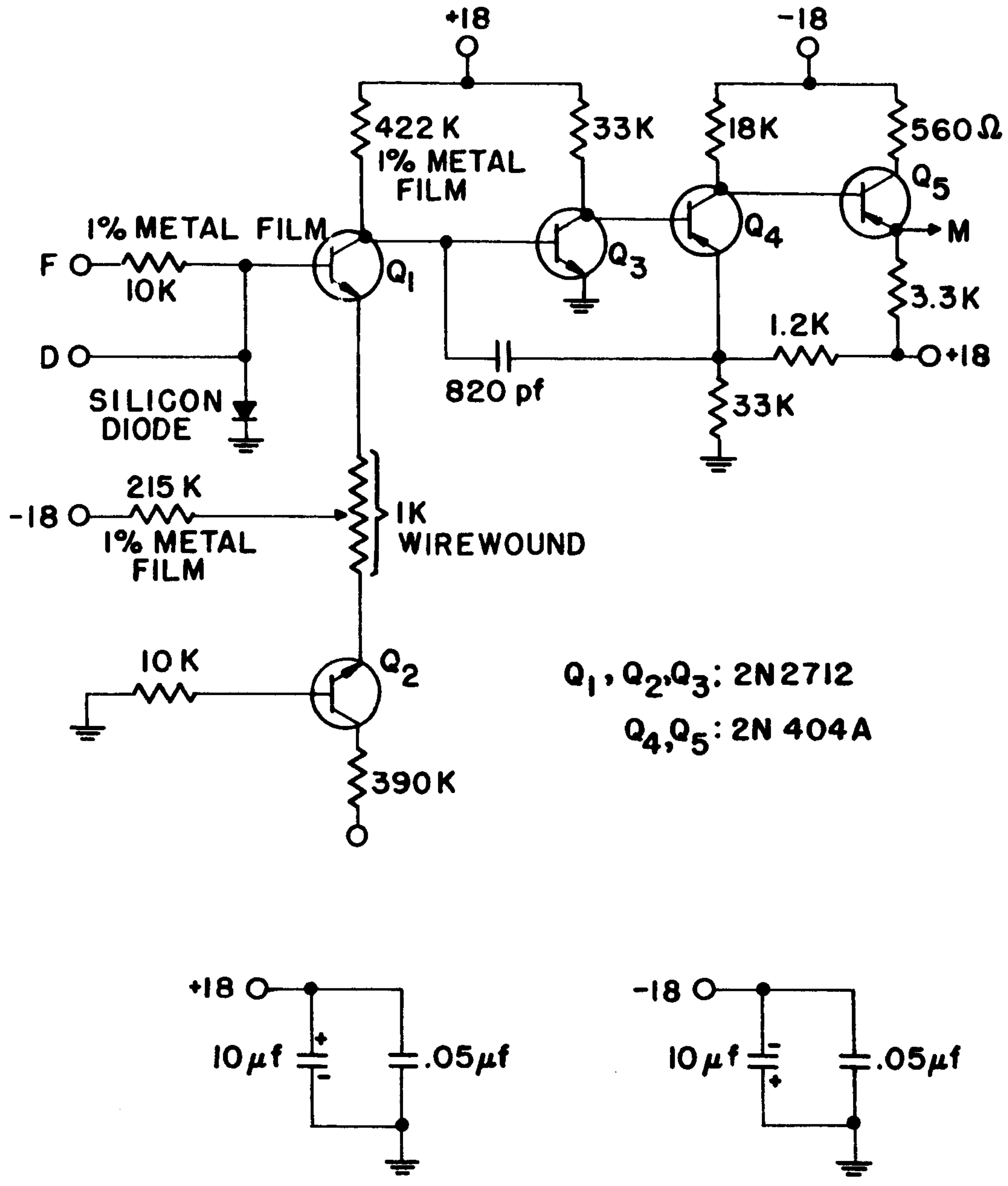


FIGURE C.1– OPERATIONAL AMPLIFIER

at 5.5 milliamperes which enables the amplifier to comfortably drive two milliamperes at $\pm$ 10 volts (corresponding to a 10K load). The full 5.5 milliamperes are not available for load current since the 3.3K emitter resistor must also be driven. Smaller loads may be driven at lower output voltages. In using the amplifier one must remember that both the feedback network and the next circuit contribute to the load.

The diode from the base of Q_1 to ground is necessary to prevent the amplifier from latching into a saturated state. It behaves as an open circuit at the small voltages that are present at the base under normal operation.

The 820 pf feedback capacitor is required to prevent high frequency oscillations. The large capacitors across the power supplies are needed to stiffen the supply voltages. Even though the power supply may have a large filter capacitor, the inductance present in a few feet of wire from the power supply to the circuit can cause ringing and possibly instability.

In using the amplifier, input D should be used as the summing point. Input F merely serves to provide a convenient summing resistor. The 1K potentiometer is used to zero the output at ground.

APPENDIX D
ZERO-ORDER HOLD

The zero-order hold network consists of a flip-flop register (rH) which drives a digital to analog converter as shown in Figure D.1. The digital input is transferred into this register in parallel. A "1" to "0" transition is first applied to clear rH to all "ones" and then the transfer is executed. The DC set and reset inputs, S_H and R_H, are used to perform the transfers $+ 127 \rightarrow rH$ and $- 128 \rightarrow rH$ respectively, if the compensator saturates.

The digital to analog converter is shown in Figure D. 2. This converter must be operated with a 12K load to ground (or to the virtual ground at the input of an operational amplifier) in order to maintain the proper bias and impedance levels.

Except for a different bias level, this decoder is identical to the one analyzed in reference 29. The resultant expression for the output voltage is:

$$e_0 = 13 - \frac{RI}{3.\,2^6} \left(d_7\, 2^7 + d_6\, 2^6 + \ldots + d_0\, 2^0\right) \quad (D.\,1)$$

where d_0 is the least significant bit (LSB) and d_7 is the sign bit of the digital input. The converter will automatically produce opposite polarity voltage swings with plus and minus inputs with two's complement coding, provided positive numbers are

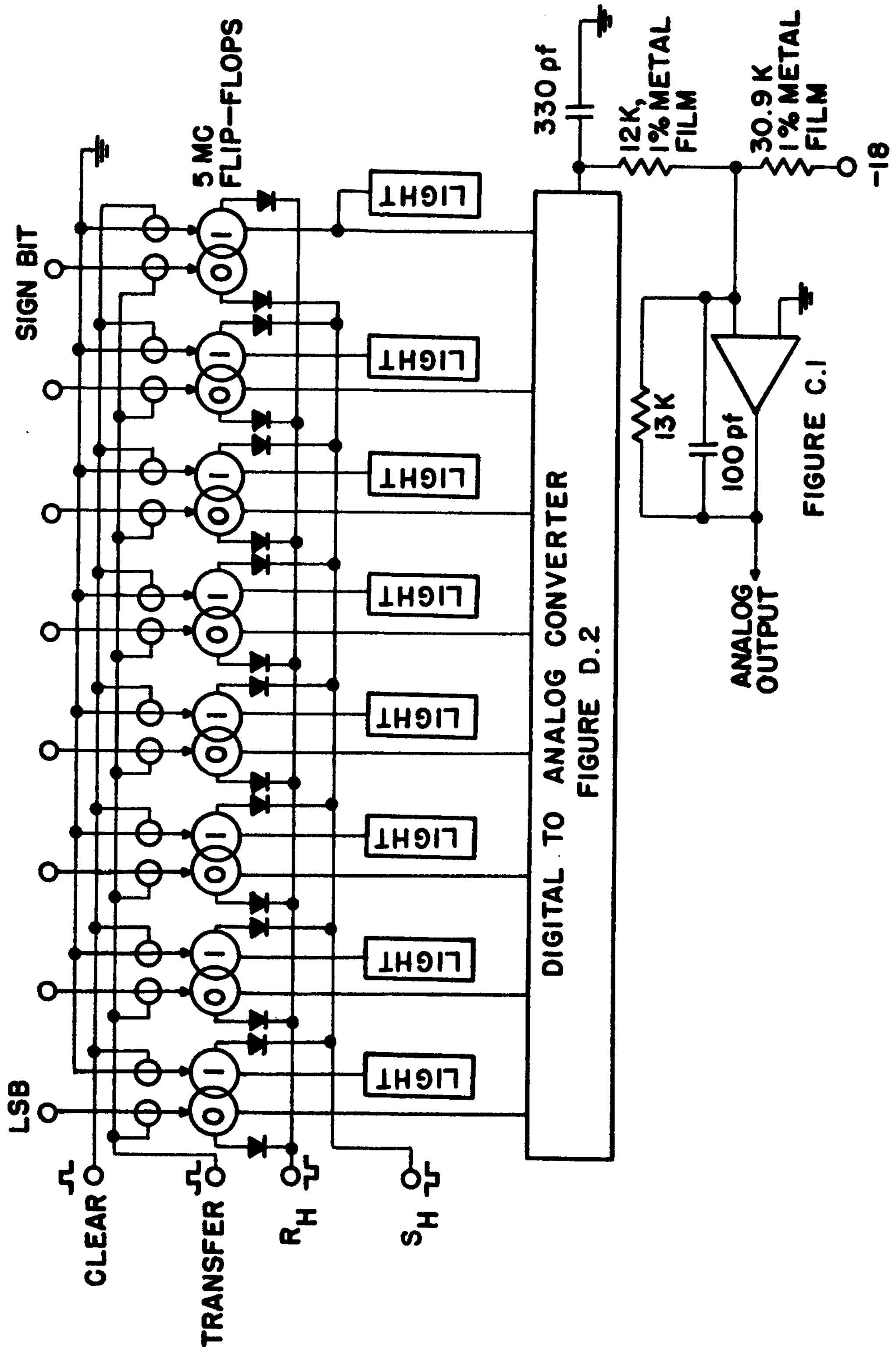
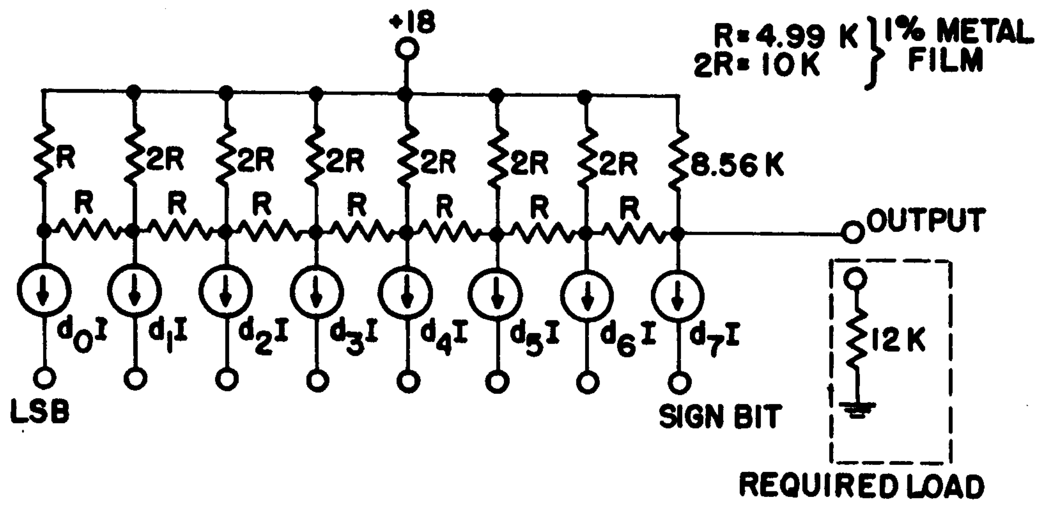


FIGURE D.1— ZERO ORDER HOLD



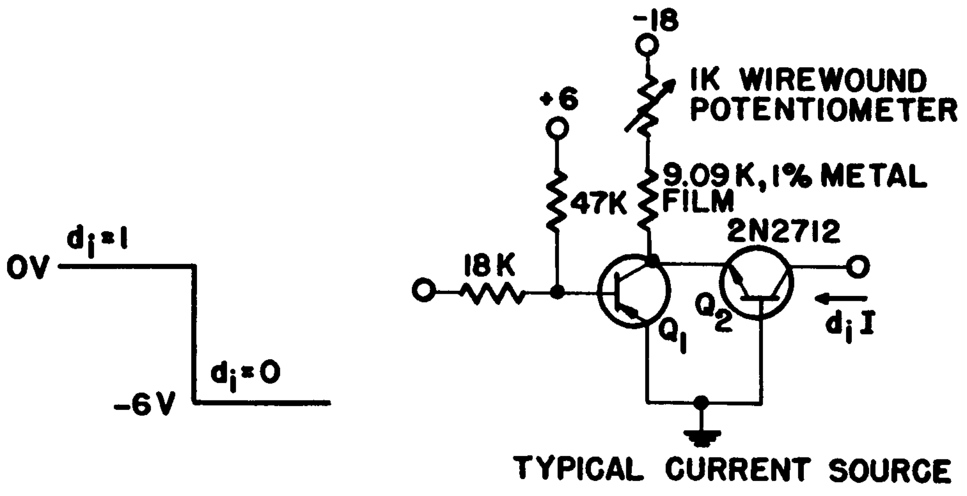


FIGURE D.2– DIGITAL TO ANALOG CONVERTER

represented by "one" in the sign bit. A digital input of 0 (represented by 1000000) then results in the output:

$$e_0\Big|_{0\ \text{input}} = 13 -(2/3)\ \text{RI}. \tag{D.2}$$

This is the DC level about which the output will vary. A digital input of +1 (represented by 10000001), for example, yields

$$e_0\Big|_{+1\ \text{input}} = (13 -\tfrac{2}{3}\ \text{RI}) - \tfrac{2}{3}\,\text{RI}\ (1/2^7), \tag{D.3}$$

while an input of -1 (represented by 01111111) yields

$$e_0\Big|_{-1\ \text{input}} \begin{aligned} &= 13 - \tfrac{2}{3}\ \text{RI}\ \left(\frac{2^7-1}{2^7}\right) \\ &= (13 - \tfrac{2}{3}\ \text{RI}) + \tfrac{2}{3}\ \text{RI}\ \left(\frac{1}{2^7}\right) . \end{aligned} \tag{D.4}$$

It is obvious that one quantum is represented by $\frac{1}{2^7}$ ($\frac{2}{3}$ RI) volts and that an inversion is present. (A positive input results in a negative output swing.) This inversion is used to good advantage in the analog to digital converter (Appendix E). There is no net inversion in the hold network, however, since the D/A converter is followed by an analog amplifier which yields another inversion. The amplifier is used to lower the output impedance of the circuit and to remove the seven volt DC level on the output of the D/A converter.

In equation (D.1), a flow of current has been associated with a binary "one". Examination of the current source in Figure D.2 will show that an output current will flow in the grounded base transistor (transistor Q_2) when Q_1 is cut off, i.e., when the input is at zero volts. Binary "one" must therefore be identified with zero volts. Since this is opposite to the convention that has been established for the input data, the 0 side of rH must be used to drive the converter except for the sign bit, because the sign bit convention has also been defined opposite to the input data.

The current sources supply a nominal current of 1.73 milliamperes. Each source must be tuned with the 1K potentiometers to provide a 50 millivolt per quanta output over the entire input range. The procedure to be followed is explained in detail on pages 54-57 of reference 1. After this has been done, any offset voltage on the output may be removed with the bias adjustment in the amplifier.

A photograph of the zero-order hold module may be seen in Figure D. 3.

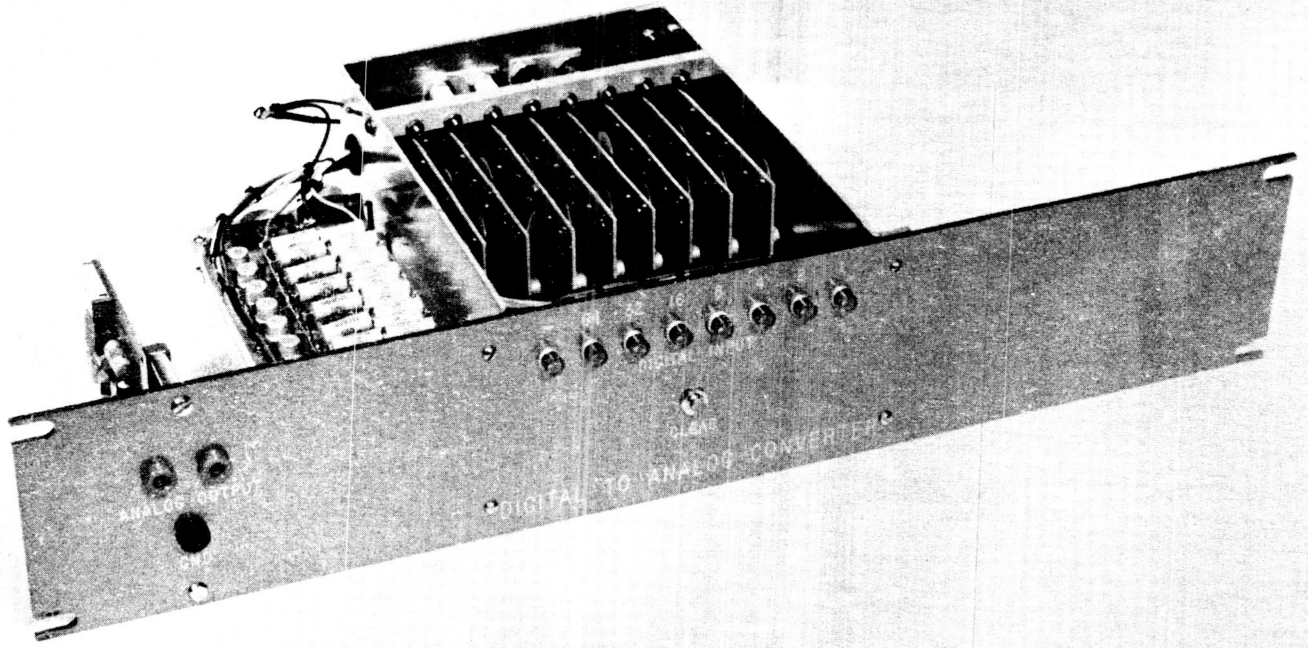


FIGURE D.3— PHOTOGRAPH OF THE ZERO—ORDER HOLD MODULE

APPENDIX E
ANALOG TO DIGITAL CONVERTER

A converter which continuously tracks the analog input was designed to convert the analog error to digital form. The maximum input is $\pm$ 6.35 volts corresponding to $\pm$ 127 quanta at 50 millivolts per quantum. The digital output is in two's complement representation.

The basic configuration is shown in Figure E. 1. The analog representation of the number stored in the bidirectional counter is continuously compared with the analog input. If these two voltages are equal to within $\pm \frac{1}{2}$ LSB (Least Significant Bit), the count input to the counter is inhibited. If the analog input is larger than the counter contents, the clock counts the counter forward; if smaller, the counter is counted backward.

This type of converter was selected mainly because the digital output appears in a bidirectional counter. The output may therefore be read by inhibiting the clock and accumulating the number of pulses required of an external clock to return the counter to zero. This is exactly the type of readout required by the digital compensator; the counter is counted down with the delay line clock as "ones" are simultaneously inserted into the serial memory.

This type of converter is not popular commercially since the conversion time is large if the analog input is suddenly

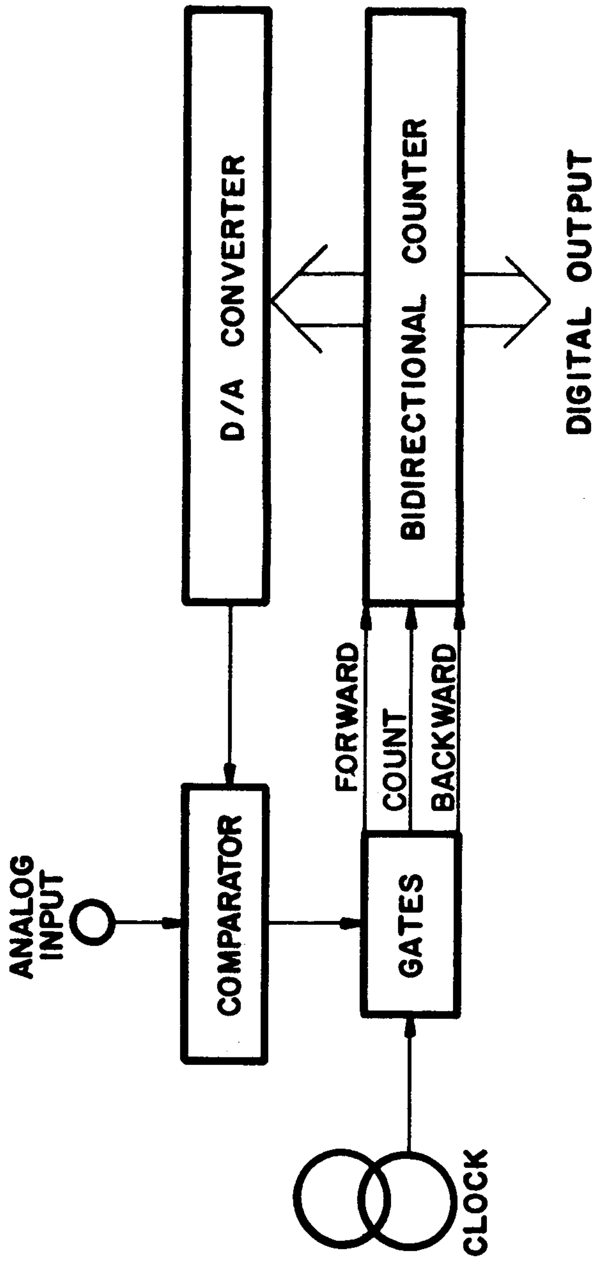


FIGURE E.I– CONTINUOUS ANALOG TO DIGITAL CONVERTER changed by N quanta. $N\tau$ seconds are required to count the counter up to the new value, where τ is the clock period. It is also obvious that the converter cannot follow an input which changes faster than $1/\tau$ quanta/second. These are not severe disadvantages in the case at hand, however, since the compensator will not interrogate the converter until at least 1.92 milliseconds have elapsed since the last sample. The converter, therefore,has 1.92 milliseconds to count from zero to the value of the analog input. In this case, τ is 8 microseconds and the maximum analog input is 127 quanta. Only 1.016 milliseconds are required to encode the maximum input.

The converter does suffer somewhat when step inputs are applied to the closed loop. If a step change is applied just before the converter is interrogated, insufficient time may elapse for conversion. The rise time of the converter, however, is small as compared to the rise time of a fast 20 cycle per second servomechanism. In response to a step input of 127 quanta, an analog system with a dominant time constant of 8 milliseconds (corresponding roughly to a 20 cycle per second break frequency) will only change about 127 (.63) = 80 quanta in 8 milliseconds or 10,000 quanta/second. The rise time of the converter is $\frac{1}{8\times10^{-6}}$ = 125000 quanta/second. The converter rise time is therefore small compared to the rise time of a 20 cycle per second servo. For step inputs of less than thirty quanta, the converter response time is truly negligible.

The converter is shown in detail in Figure E. 2. The two combinational circuits and the functions they realize are shown in Figures E. 3 and E. 4. The BDC (Bidirectional Counter) has two trigger inputs, C_1 and C_2, and is identical to the counter described in Appendix B except that it has eight stages.

The output of the comparator consists of two binary outputs which define whether the BDC is to count forward ($F_c = 1$, $B_c = 0$); backward ($F_c = 0$, $B_c = 1$); or not at all ($F_c = B_c$). Since these levels may change at any time, they are transferred (at the leading edge of the clock pulse) into two buffer flip-flops two microseconds before the BDC is to be incremented. This ensures that the forward (F) and backward (B) levels for the BDC do not change during the two microseconds preceding a count input. These levels must not be allowed to change during the two microseconds preceding a count input since this much time must be allowed for the levels within the counter to stabilize (see Appendix A).

The outputs of the buffer flip-flops are also used to inhibit the BDC input trigger whenever $F_B = B_B$. Equality is used as the inhibiting criteria instead of the simpler condition: $F_B = 0$, $B_B = 0$. Then if noise on the inputs to the comparator should ever result in the forbidden state: $F_B = 1$, $B_B = 1$, the counter will not be incremented. The trigger is also inhibited

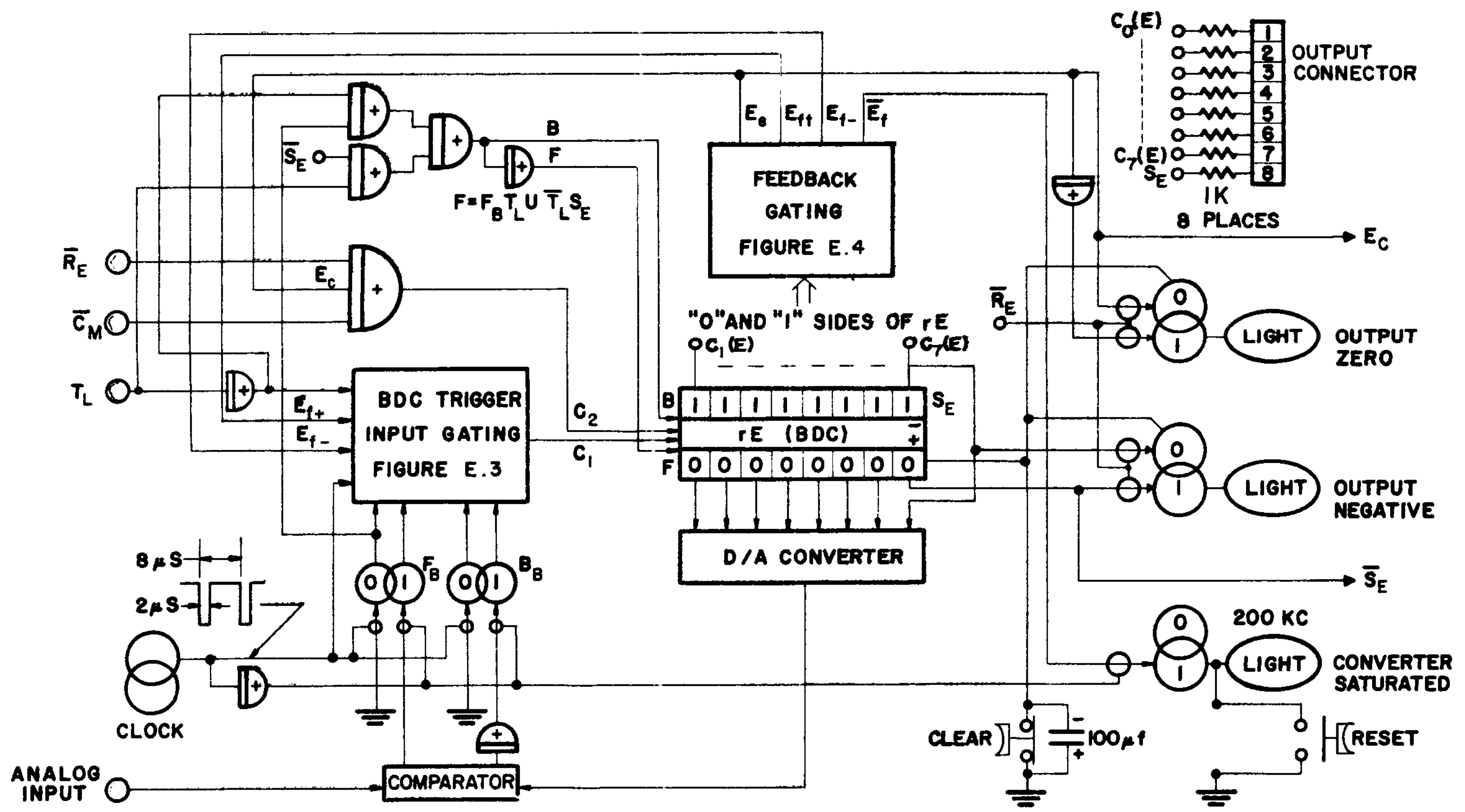
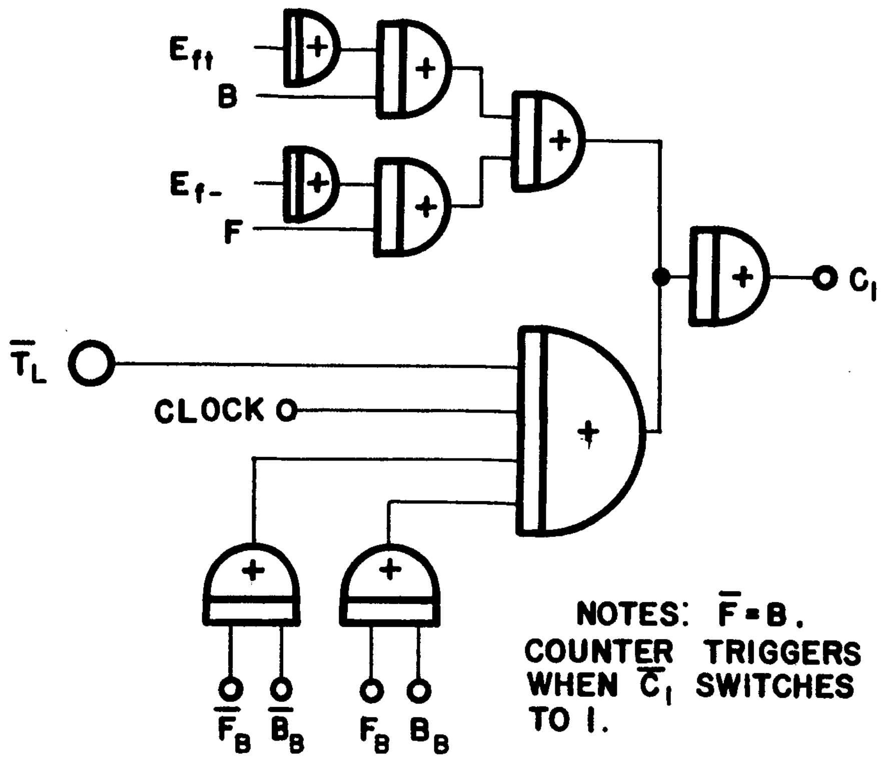


FIGURE E.2 — ANALOG TO DIGITAL CONVERTER DETAILS



NOTES: $\bar{F} = B$.
COUNTER TRIGGERS WHEN $\bar{C}_1$ SWITCHES TO 1.

$$\bar{C}_1 = T_L\ \overline{(\text{CLOCK})}\ (\overline{F_B B_B \cup \bar{F}_B \bar{B}_B})(\overline{E_{f+} F \cup E_{f-} B})$$

FIGURE E.3— BDC TRIGGER INPUT GATING

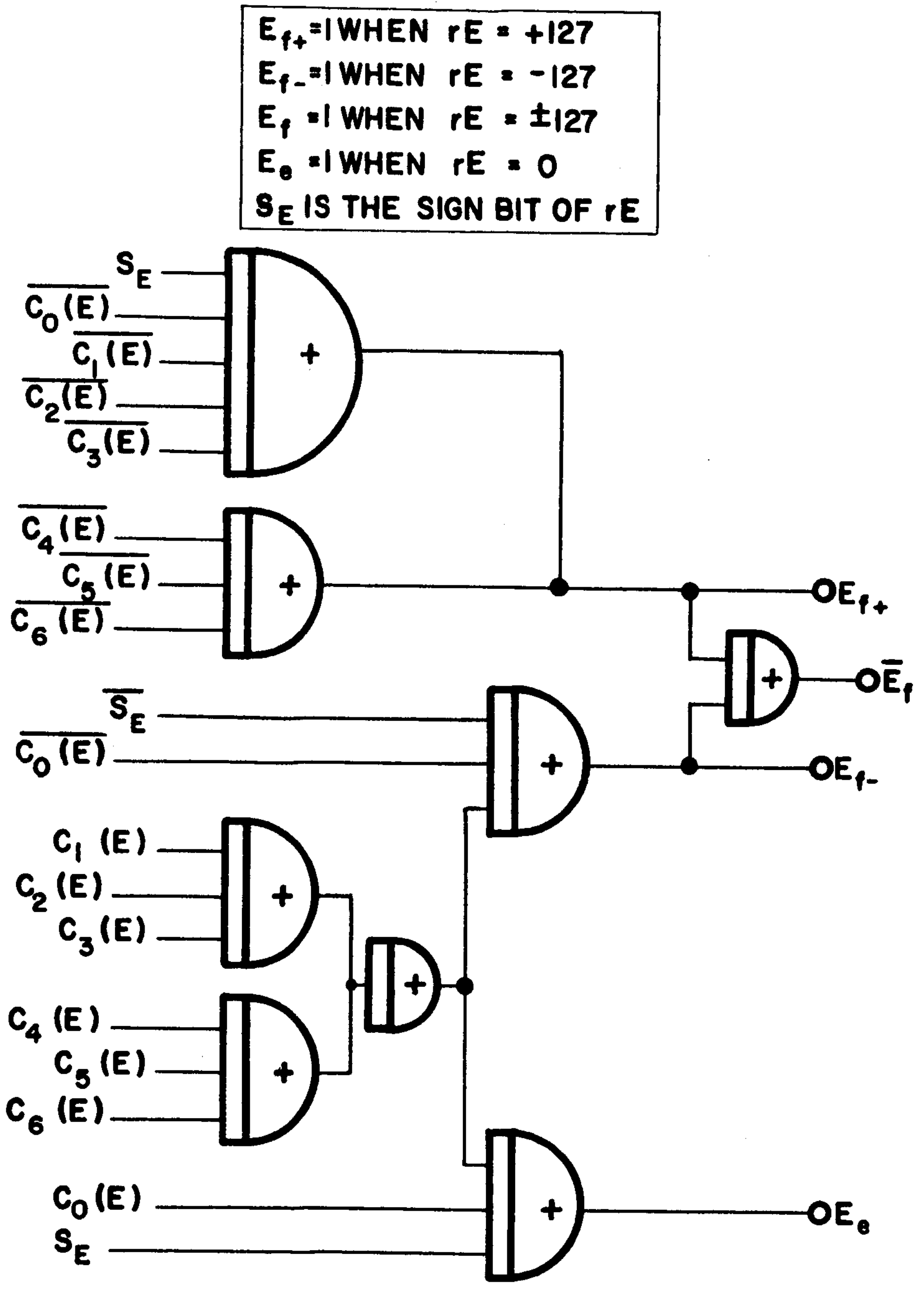


FIGURE E.4— FEEDBACK GATING

whenever the contents of the counter tends to exceed $+127$ ($F\ E_{f+} = 1$) or -127 ($B\ E_{f-} = 1$).

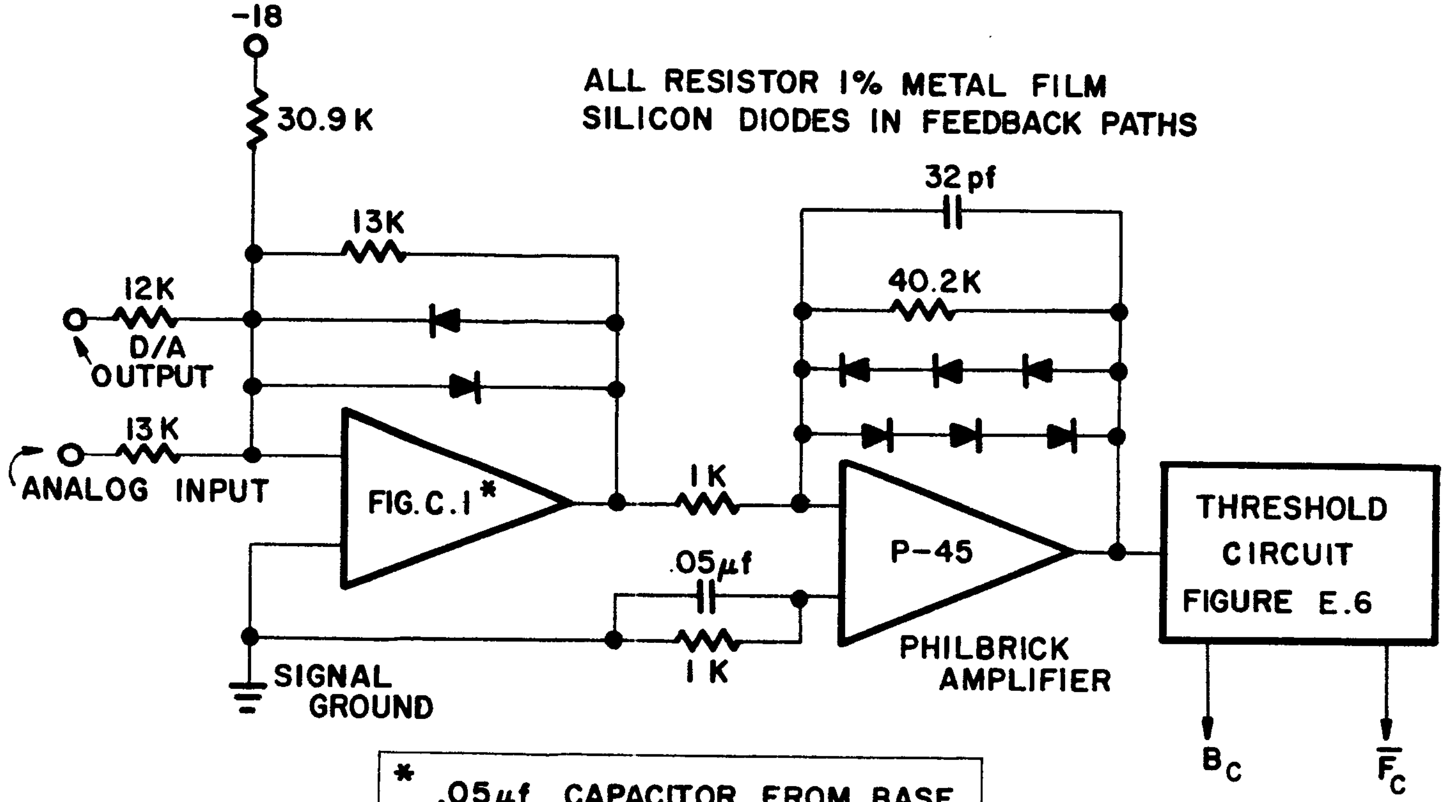
The two input levels T_L and $\overline{R}_E$ define whether the converter is in the tracking mode or read mode. When T_L (track command level) is true and R_E (read level) is false, the converter will continuously track the analog input. When $T_L = 0$ and $R_E = 1$, the BDC will be counted to zero by the external clock $\overline{C}_M$. If $T_L = 0$ and $R_E = 0$, the contents of the BDC will remain constant; the input $T_L = 1$, $R_E = 1$ is not allowed. When switching from the track mode to the read mode, the level T_L must be changed at least two microseconds before R_E is changed. Again, this must be done to ensure that the forward and backward levels into the BDC stabilize two microseconds before a count input is received.

A "one" to "zero" transition of the internal clock triggers the BDC with a "one" to "zero" transition of the counter input C_1 (provided the other levels are satisfactory) and resets the two buffer flip-flops. This prepares the flip-flops for accepting the comparator output at the leading edge of the next multivibrator pulse and performs one other important function: resetting the buffer flip-flop returns the counter input C_1 to "one" and holds it there. This occurs before any static or dynamic hazards in the feedback gating can cause erroneous triggering of the BDC.

The usual trailing edge triggering is used for the other trigger input. The control level $\overline{R}_E$ must therefore be changed while $\overline{C}_M$ equals 1 in order to prevent false triggering of the counter. This type of triggering could not be used for the other input since the control level T_L is not synchronized with the internal clock of the converter. Changes in this level could therefore trigger the counter.

Three flip-flops and lights were used to indicate various states of the converter. The two flip-flops which are triggered by the read command tell whether the output is zero, positive, or negative at the time the converter is interrogated. The other flip-flop is triggered by the internal clock and indicates whether the output ever reaches saturation ($\pm$ 127).

The comparator is shown in Figure E. 5. Since there is an inversion through the D/A converter, the comparison is performed by summing its output with the analog input. This is done with the first amplifier. The 30.9K resistor to -18 volts is used to remove the bias level present on the D/A output. The input and output of the second amplifier then vary about zero volts. The second amplifier is a wide-bandwidth amplifier which is used for gain only. The threshold circuit detects when the amplifier output exceeds $\pm$ 1 volt. Since a difference of $\pm$ 25 millivolts ($\pm \frac{1}{2}$ LSB) must be detected, a gain of 40 is needed.



* .05μf CAPACITOR FROM BASE OF Q_2 TO GROUND WAS ADDED TO INCREASE BANDWIDTH.

FIGURE E.5– COMPARATOR CIRCUIT

It may appear that one amplifier could be used to perform both the addition and gain functions. However, the George A. Philbrick P-45 amplifier does not have the necessary gain-bandwidth product if its input resistor is increased to 12 K as required by the D/A converter. The amplifier of Figure C-1 does have the necessary bandwidth at this impedance level when operated at unity gain.

The silicon diodes are used to keep the amplifiers out of saturation and hence increase the comparison speed. The diodes appear as open circuits over the voltage range of interest. If the signals tend to become large, however, the gain of the amplifiers is automatically reduced, limiting the output.

The threshold circuit is shown in Figure E. 6. The emitters of transistors Q_1 and Q_2 are biased at - 0. 9 and + 0. 9 volts, respectively, by the silicon diodes. These transistors are therefore cut off, as are the output transistors, while the input lies within $\pm$ 0. 9 volts. When this voltage range is exceeded, one of the transistors will conduct and drive current through the tunnel diode which is connected to its collector. When the peak current of the tunnel diode is exceeded (at about one volt input), it switches to its high voltage state, saturating an output transistor. Thus, three distinct states are obtained for controlling the BDC.

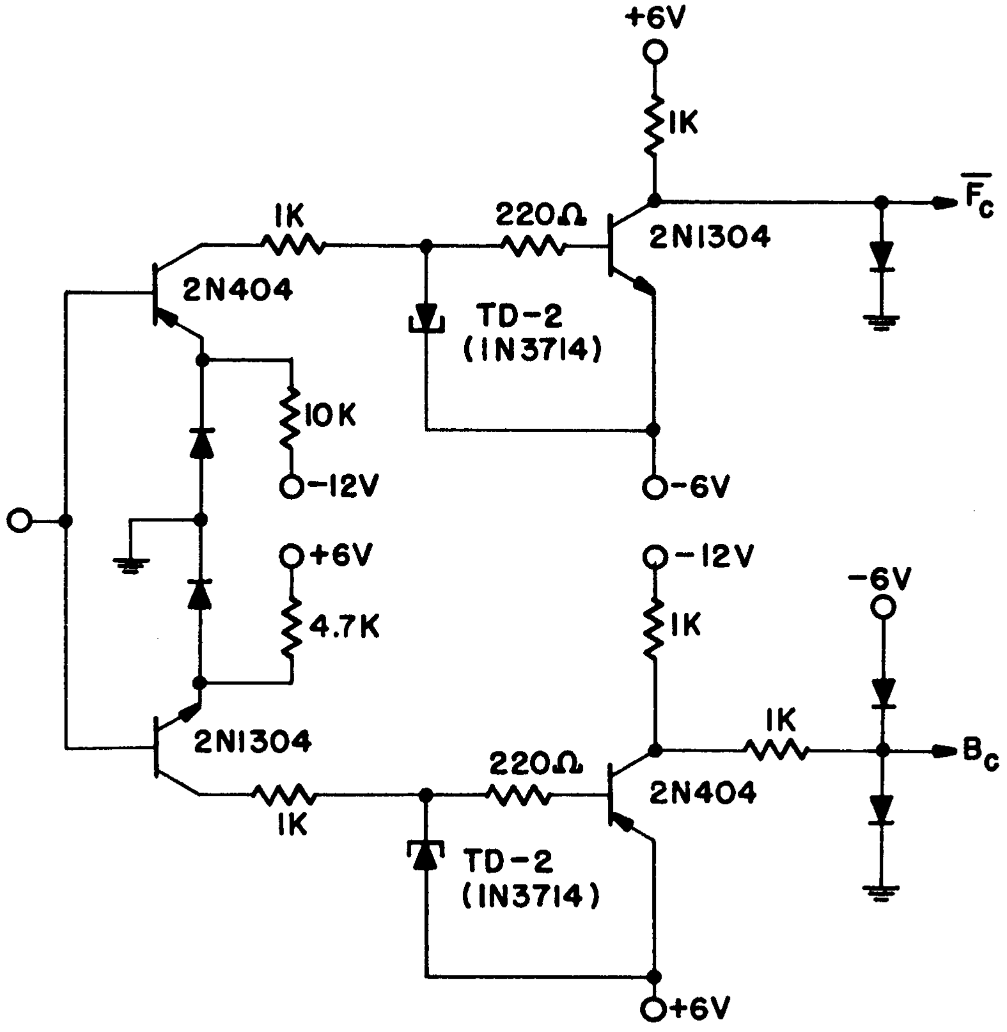


FIGURE E.6 — THRESHOLD CIRCUIT

When large transients are introduced by switching the D/A converter, four microseconds are required for the amplifiers to stabilize. Two microseconds were allowed for the threshold circuit to stabilize. Two more microseconds were needed for the forward-backward levels in the BDC to stabilize. These time delays account for the eight microsecond clock period.

The clock is shown in Figure E. 7. Figure E. 8 is a photograph of the A/D converter.

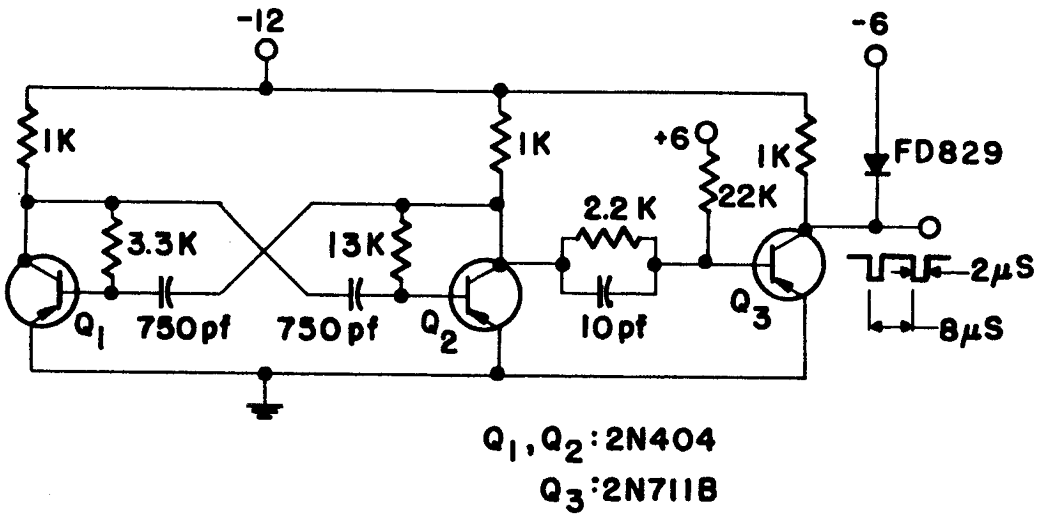


FIGURE E.7— A/D CONVERTER CLOCK

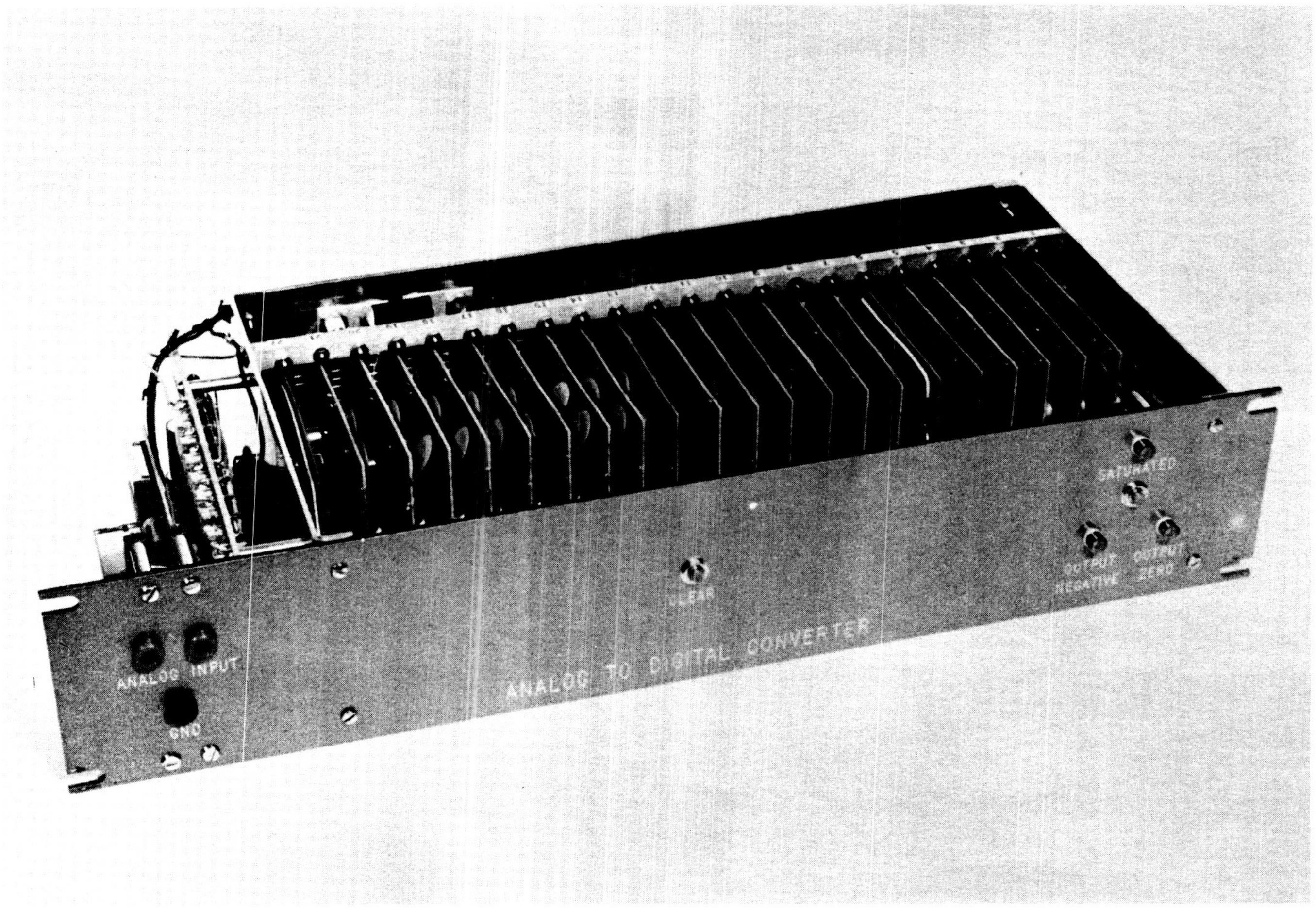


FIGURE E.8— PHOTOGRAPH OF THE ANALOG TO DIGITAL CONVERTER